

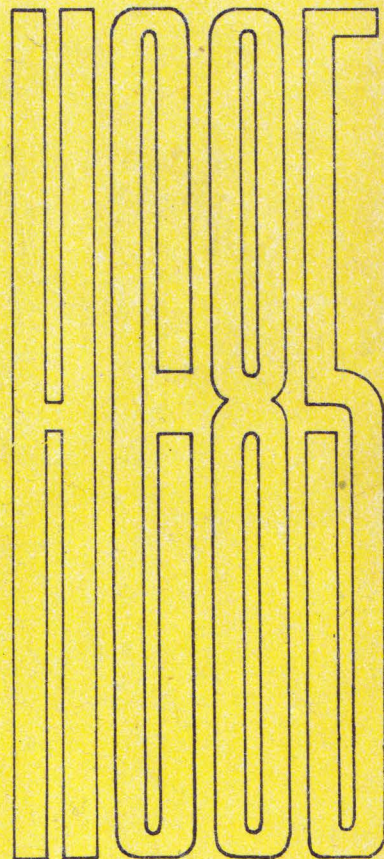
Gh. Rizescu  
P. Zamfirescu  
E. Dobrovie

A. Petrescu  
N. Țăpuș  
V. Cososchi  
N. Badea

T. Moisa  
M. Mârșanu  
C. Hărăbor

# abc de **CALCULATOARE PERSONALE** și... nu doar atât...

# 2



EDITURA TEHNICĂ







27.09.1994  
Shore

AUTOMATICA  
INFORMATICA  
ELECTRONICA  
MANAGEMENT



SERIA INITIERE



## **BIBLIOTECA DE**

### **Automatică — Informatică — Electronică — Management**

#### **SERIA ÎNȚIERE**

- E. VASILIU**  
**ÎNȚIERE ÎN DISPOZITIVELE SEMICONDUCTOARE**
- D. STANOMIR**  
**ÎNȚIERE ÎN ELECTROACUSTICA**
- W. TRUSZ**  
**ABC-UL REPARĂRII RADIORECEPTOARELOR**  
Traducere din lb. polonă (Ciclul ABC-uri)
- A. POPA**  
**ABC DE PROTECȚIA MUNCII** (ciclul ABC-uri)
- MARGARETA DRĂGHICI**  
**ÎNȚIERE ÎN COBOL**
- STELIAN NICULESCU**  
**ÎNȚIERE ÎN FORTRAN**
- PAUL CONSTANTINESCU ȘI ZAHARIA NICOLAE**  
**ÎNȚIERE ÎN ORGANIZAREA ȘI PROIECTAREA SISTEMELOR DE CONDUCERE**
- I. CREȚU**  
**ÎNȚIERE ÎN ESTETICA PRODUSELOR** (ciclul ABC-uri)
- E. AISBERG**  
**ABC DE RADIO ȘI TELEVIȚIUNE**  
Traducere din limba franceză
- J. D. WARNIER, B. MI FLANGAN**  
**ÎNȚIERE ÎN PROGRAMARE**  
Traducere din limba franceză
- I. H. BERNHARD, B. KNUPPERTZ**  
**ÎNȚIERE ÎN TRISTOARE**  
Traducere din limba germană
- W. DEPPEERT, K. STOLL**  
**ÎNȚIERE ÎN PNEUMOAUTOMATICA**  
Traducere din limba germană
- ÎNȚIERE ÎN RADIOELECTRONICA CUANTICA**
- V. POPESCU**  
**ÎNȚIERE ÎN PROGRAMAREA ÎN CALCULATOARE NUMERICE**
- I. PAPADACHE**  
**AUTOMATIZĂRI INDUSTRIALE, ÎNȚIERE, APLICAȚII**
- ȘT. NICULESCU**  
**FORTRAN, ÎNȚIERE ÎN PROGRAMARE STRUCTURATA**
- J. FORRESTER**  
**PRINCIPIILE SISTEMELOR: TEORIE ȘI AUTOÎNȚIERE**  
**PROGRAMATA**  
Traducere din lb. engleză — S.U.A.
- P. DRANSFIELD, D. F. HABER**  
**ÎNȚIERE ÎN PROGRAMAREA ÎN METODA LOCULUI RĂDACINILOR**  
Traducere din lb. engleză — S.U.A.
- D. RODDY**  
**ÎNȚIERE ÎN MICROELECTRONICA**  
Traducere din lb. engleză
- NICULESCU CL., IOSIF M.**  
**ÎNȚIERE ÎN COMUNICAȚIILE PRIN FIBRE OPTICE**
- CSABAI DĂNIEL**  
**TEHNICA SONORIZĂRII** (traducere din lb. maghiară)
- MITROFAN GH., PFLANZER G.**  
**ÎNȚIERE ÎN TELEVIȚIUNEA ÎN CULORI**
- RADU NEGOESCU**  
**ÎNȚIERE ÎN ELECTRONICA BIOMEDICALA**
- RADU NEGOESCU**  
**INSTRUMENTAȚIA ELECTRONICA BIOMEDICALA**
- A. PETRESCU s.a.**  
**TOTUL DESPRE... CALCULATORUL PERSONAL aMIC**
- J. DUMITRAȘCU s.a.**  
**ÎNȚIERE ÎN FORTRAN... CONVERSIND CU CALCULATORUL**
- L. DUMITRAȘCU**  
**ÎNȚIERE ÎN COBOL... CONVERSIND CU CALCULATORUL**
- L. DUMITRAȘCU**  
**ÎNȚIERE ÎN MICROELECTRONICA INTERACTIVA**
- M. PATRUBANY**  
**TOTUL DESPRE... MICROPROCESORUL Z 80**
- L. DUMITRAȘCU**  
**ÎNȚIERE ÎN MICROELECTRONICA INTERACTIVĂ**
- M. PATRUBANY**  
**TOTUL DESPRE... MICROPROCESORUL Z 80**



Prof. dr. ing.

**Adrian Petrescu**

prof. emerit liceu mat.

**Gheorghe Rizescu**

șef catedră dr. ing.

**Nicolae Țăpuș**

șef lucrări dr. ing.

**Trandafir Moisa**

drd. ing.

**Paul Zamfirescu**

ing.

**Victor Cososchi**

drd. ing.

**Mihai Mârșanu**

ing.

**Eugen Dobrovie**

drd. ec.

**Nicolae Badea**

prof. liceu. mat.

**Constantin Hărăbtor**

# abc de CALCULATOARE PERSONALE și... nu doar atît...

Coordonare generală: Prof. dr. ing. **ADRIAN PETRESCU**

**Volumul 2**

**EDITURA TEHNICĂ**

București, 1990



## CONTRIBUȚIA AUTORILOR

**A. Petrescu:** 13 p, 14, 16. 3, 19 p, 23, 24, 25.1 p, 25.2, P-D-E p  
**Gh. Rizescu:** 15.1, 15.2, 15.3, 15.4, 15.5, 17,18, 20, 21, 22, A 1., A 5  
**N. Țăpuș:** 16.5 p, 19 p, 25.1 p, A 3; A 4p.  
**T. Moisa:** 25.1 p  
**P. Zamfirescu:** 16.1, 16.2, 16.4, 19 p, P-D-E (p)  
**M. Mârșanu:** 15.6 p, 16.6 p, 16.5p, 18.41, 18.42, A 6  
**E. Dobrovie:** P-D-E p  
**V. Cososchi:** P-D-E p  
**N. Badea:** P-D-Ep, A 2, A 4p.  
**C. Hărăbor:** 15.6 p, 16.5 p.

RECENZII: Dr. ing. **PAUL CONSTANTINESCU**, dr. ing. **LIVIU DUMITRAȘCU**  
dr. ec. **GHEORGHE SABAU**, ing. **CONSTANTIN MINDRULEANU**

REDACTOR: ing. **PAUL ZAMFIRESCU**

**Mulțumiri întregului colectiv al Întreprinderii poligrafice Sibiu (Director:** ing. Petre Pascu; ing. șef: Gheorghe Tămaș; producție: ing. Ioan Șiandru (șef producție), Eugenia Oleksik; **maiștri în diverse compartimente:** Ioan Simțion (m. princ.), Ioan Stroie, Vasile Ienciu, Iancu Popescu, Ioan Rodean, Mircea Pascu, dar mai ales, celor care — cu deosebire — au contribuit direct la realizarea lucrării: **(Tastere monotyp:** Marioara Suciu, Emilia Lungu, Livia Mihu; **monoturnare:** Mihai Schuster, Ioan Gîndilă, Ilie Lupu, Ionel Bica; **linotip:** Florentina Murărescu, **paginare:** Dan Koss, ajutat de Mioara Mașca, Ana Berea, Ana Străjan; **pregătire offset:** Nagy Sibile, Fânica Popovici, Ileana Pădureanu, Marcela Hiesch, Daniela Boantă, Cornelia Spinei, Mariana Băilă, Victoria Barbulat, Mirela Popa; **tipar offset:** Nicolae Prișcă, Nicolae Vinersar, ajutați de Ioan Funariu, Octavian Moga; **corectură:** Marcela Tamaș, Alina Iliu, Maria Cheleț, Maria Stoisor; **legătorie:** Jenica Matei, Elena Lăcătuș, Ana Szylaghi, Paraschiva Giuscă, Ștefan Nanași, Cornel Banea  
Cu scuzele de rigoare pentru omisiunile nedorite, reparabile în al doilea tiraj.

ISBN 973-31-0013-7

ISBN 973-31-0015-3

COPERTE: Arh. **SILVIA PINTEA**  
DESENE: Arh. **IULIAN MIHAESCU**

TEHNOREDACTARE: (inițială) **VICTORIA UNGUREANU**;  
adaptări și completări 1990 — redacția

Bun de tipar 27.04.1990; Coli de tipar 23,5.

Cărțile s-au realizat la Întreprinderea poligrafică Sibiu



27.08.1984  
Stancu

## Cuprins general

### Volumul 1

Studiu introductiv (Acad. N. Teodorescu) .....	V
Cuprins general vol. 1 și vol. 2 .....	XI
PROLOG — DIALOG — EPILOG (continuă în vol. 2) .....	XIII
Cuprins detaliat vol. 1 .....	XXVIII
Partea I — Calculatoare, microcalculatoare și calculatoare personale în țara noastră și pe plan mondial .....	1
Partea II — Calculatoare numerice. Realizare fizică. Baze aritmetice și logice .....	37
Partea III — Calculatorul personal HC-85. Structură, componente, operare, programare .....	66
Partea IV — Programarea în limbașul BASIC pe calculatorul HC-85 .....	103
Partea V — Programarea în limbașul LOGO pe calculatorul HC-85 .....	175
	— 312

### Volumul 2

Cuprins general vol. 1 și vol. 2 .....	V
PROLOG — DIALOG — EPILOG — (continuare din volumul 1) .....	VI
Cuprins detaliat vol. 2 .....	XIV
Partea VI — Microcalculatorul HC-85 în procese industriale. Pachete de programe de aplicații pentru HC-85 .....	1
Partea VII — Calculatorul personal în învățământ și educație .....	44
Partea VIII — Programe educaționale în BASIC pe calculatorul HC-85 .....	111
Partea IX — Microbiblioteca de programe pe casete .....	182
Partea X — Complemente matematice .....	192
Partea XI — Complemente informatice .....	236
BIBLIOGRAFIE GENERALĂ, CONSULTATĂ ȘI RECOMANDATĂ .....	275
ANEXE — 1. Metodica algoritmi; 2. Programe educaționale; 3. Interpretorul BETA BASIC; 4. Concursuri informatică (probleme rezolvate) (continuă din vol. 1) 5. Probleme în pseudocod; 6. Soluții ale temelor din 16.6 .....	279
PROLOG — DIALOG — EPILOG (continuare din vol. 2); HC-85 extins, HC-88 disc, rețea .....	320 —
	— 360

# PROLOG — DIALOG — EPILOG

(continuare din volumul 1)

Îmi place la Sorbona sala Descartes. În sala „ideilor clare și distincte“ o frescă pe întreg zidul din fund arată câteva nimfe ieșind din abur spre lumină și formă. Sunt, parcă, *ideile* din poemul „Aurore“ al lui Paul Valéry: Quoi! c'est vous mal deridées / Que fîtes-vous, cette nuit / Maitresses de l'ame, Idées/..\*.

CONSTANTIN NOICA

După ce am prezentat în aceeași secțiune din primul volum cîte ceva despre calculatoarele personale românești, unele date — pînă în 1989 — privind informatizarea învățămîntului din țară, stadiul actual al informatizării învățămîntului mondial, cum și cîte ceva despre carte și despre autori, ne vom referi atît la mijloace și metode disponibile, la inițiative 1990 ale autorilor și editorului, cît și la măsuri și programe ale noului Minister al Învățămîntului, în legătură cu informatizarea învățămîntului românesc, în noile condiții generate de Revoluția din decembrie 1989.

## Cîte ceva despre mijloace și metode disponibile

● Nu vom mai insista asupra unităților de calcul independente de tip HC-85, despre care am vorbit în decursul lucrării; nu ne vom referinici la familiile de calculatoare Felix PC (compatibile IBMPC) sau la cele superioare acestora, de exemplu PS/2, care au făcut obiectul capitolului 25. Vom reflecta mai pe larg evoluția familiei HC-85, prin configurațiile HC-85 extins și HC-88, (v. în continuare) utilizabile cu unități de discuri flexibile și în rețele locale. Vom porni, însă de la o aplicație, **produsul autonom INFOBUS**, care este un laborator mobil cu o rețea locală de calculatoare, proiectat în 1988 în România de către specialiști din actualul Institut de Cercetări pentru Informatică (ICI). Laboratorul INFOBUS este, cum ne sugerează și numele un autobuz în care sînt asigurate condiții pentru un proces modern de instruire în informatică, avînd la bază un **complex de calculatoare personale, mijloace pentru transmiterea de imagini video și semnale audio**, produse-program educaționale și profesionale. INFOBUS permite practicarea unei **tehnologii didactice eficiente**, ce îmbină instruirea individualizată și instruirea în grup, în care profesorul este direct implicat. Pentru asigurarea mobilității se folosește un autobuz de 11/2,5/3 m, cu motor de 192 CP/2100 rotații pe minut; spațiul de lucru este de 8×2,4 m<sup>2</sup>, ca laborator pentru 10+1 posturi de lucru, respectiv pentru 20+1 persoane. **Tehnica de calcul** este constituită din 11 calculatoare personale HC-85 extins, cuplate într-o rețea specializată; **unul dintre calculatoarele HC-85 extins (sau HC-88) este dotat cu unitate de discuri flexibile, monitor color și interfață de comunicație pentru postul de lucru central**; celelalte 10 calculatoare HC-85 extins cu monitoare alb/negru și interfață de comunicație pentru fiecare dintre posturile de lucru. De remarcat: **asineronismul total între operațiile de prelucrare efectuate la posturile de lucru**; îmbinarea instruirii individualizate cu cea de grup (prin VMS); formarea unui stil de lucru profesional prin posibilitățile de a schimba dinamic informații între elev-calculator, elev-elev, elev-profesor, prin accesul la bibliotecile de programe de pe discuri prin metode simple de gestionare. Se poate asigura astfel: predarea informaticii sau a altor discipline; executarea de lucrări practice de progra-

---

\* O traducere liberă ar putea fi: Iată-vă! Somnoroaselor / Ce-ați făcut az'noapte / Stăpine ale sufletului, Ideilor/.., la care noi am putea adăuga și următorul vers „persiflant“ al poemului „Courtisanes par ennui?“ cu traducerea sa posibilă „Curtezane din plictis?“.



mare individuale sub controlul profesorului; demonstrații practice cu produse informatice; verificări de cunoștințe on-line; controlul dinamic al procesului de instruire; colectarea și prelucrarea datelor statistice din acest proces; un suport mobil pentru concursuri și „tabere” de informatică.

**Bibliotecile de programe informatice educaționale**, instalate pe unitatea de disc flexibil, sînt accesibile la fiecare post de lucru prin comenzi BASIC sau CP/M, în funcție de regimul selectat. Menționăm, ca un fapt important, că în INFOBUS pot fi instalate și alte tipuri de calculatoare, de ex. calculatoare personal-profesionale de tip Felix-PC, astfel încît INFOBUS constituie un „modul educativ computerizat, flexibil în ce privește tipurile de calculatoare folosite”.

**Sistemul de recepție-transmitere a imaginilor video** este bazat pe o monitorizare electronică a imaginilor (VMS—Video Monitoring System), proiectată special pentru INFOBUS și instalată la postul central; VMS preia imagini video de la o sursă și le transmite pe monitoarele posturilor de lucru; imaginea provine de la oricare dintre aceste monitoare sau de la surse externe, cum sînt camerele de luat vederi, video casetofonele, receptoarele TV ce primesc semnale de la stații terestre sau spațiale.

**Rețeaua locală video (LVN)** astfel formată poate conecta pînă la 16 posturi de lucru (elevi) la postul central (profesor) într-o rețea de tip magistrală, care oferă 7 canale analogice, de mare viteză, pe care se pot transmite semnale TV (R, G, B, PCL, SINCRO); **magistrala video** are lungimea totală de 40 m, **Unitatea de comandă** realizată cu microprocesor este la postul central; toate unitățile (monitor, calculator) sînt dotate cu cite un adaptor de linie. Pe monitorul postului central se pot afișa imagini de la un post de lucru selectat sau (un timp limitat și ciclic de cite 20 secunde fiecare) imagini de la toate monitoarele posturilor de lucru. Sistemul permite, deci, un **proces interactiv de instruire asistată** INFOBUS. cuprinde dispozitive de preluare (de la microfon-magnetofon) — **amplificare și transmitere a sunetelor**, de alimentare cu energie (trifazată, 220 V, 50/60 Hz).

INFOBUS este disponibil ca model și constituie — ca rețea locală educațională — un produs și cu aprecieri favorabile de peste hotare, ce poate fi util învățămîntului românesc.

● Tot din realizările ICI mai menționăm **proiectul pilot MINICOMP**, realizat îndeosebi pentru acomodarea copiilor din grădinițe (nr. 52 București), școli primare (nr. 17 București), gimnazii, cercuri, tabere ș.a. cu calculatoarele personale. S-au constituit suporturi de curs adaptate diferitelor vîrste: un **produs-program** pentru preșcolari, BAMBY (opt programe, cu comenzi simple pentru dialogul copil-calculator); **cursuri de BASIC**, diferențiate pe vîrste — cum și de BETA BASIC, LOGO, FORTH. Instructorii au constituit o echipă multidisciplinară, colaborînd și cu catedra de pedagogie și psihologie (Univ. Cluj-Napoca). În urma acestor experimente, ICI este în măsură să ofere atît **programe pentru instruirea preșcolarilor, de transmitere a primelor noțiuni de programare** (începînd cu clasa a II-a) cît și a **însușirii unor limbaje de programare** pentru clasele II—IV, V—VI și VII—VIII. Acestea se sprijină pe noua activitate autonomă a **Bibliotecii Naționale de Programe (BNP)** și pe validarea și de către profesori-pedagogi a produselor program oferite.

● Pentru a constitui o punte către propunerile de generalizare a informatizării învățămîntului (care țin și de bune legături pentru viitor cu UNESCO și cu alte organisme internaționale) ne vom referi pe scurt și la **colaborările existente cu UNESCO**, în anii trecutului regim (în condiții grele) pornind de la Aide-memoire-ul semnat între România și UNESCO în 1981. În acest an ICI a realizat un studiu „**Informatica, problemele lumii actuale și viitorul umanității**” iar în 1982 — o lucrare „**Utilizarea microcalculatoarelor la predarea matematicii și a altor discipline științifice în școli**”, aceasta din urmă este și titlul unui proiect pilot UNESCO, la care participă și institute din RFG, Anglia, Ungaria și care este coordonat din 1988 de ICI și de Universitatea București, în cadrul **Programului de cooperare al UNESCO în cercetarea și promovarea educației în țări din sudul și sud-estul Europei (CODISEE)**. În acest program se participă, în 1988, împreună cu CEPES (Centrul pentru Învățămîntul Superior al UNESCO la București) la o reuniune de experți dintr-o **rețea regională UNESCO** cu 10 țări membre.

S-a participat, cum s-a mai afirmat în carte, cu echipe la manifestările UNESCO „Copiii în era informaticii” din ultimii ani, în Bulgaria.

De asemenea prin activitatea creatoare în domeniul calculatoarelor, prin activitatea la catedre cum și în sprijinirea manifestărilor extrașcolare de informatică, ca și în viața editorială, membri ai colectivului de autori, alături de mulți specialiști din țară, și-au adus aportul prețios, unele activități reflectîndu-se, într-o măsură, în diferitele părți ale lucrării.



## Cîte ceva despre propunerile noastre 1990, despre măsuri și programe ale actualului Minister al Învățămîntului

Societatea, îndeosebi în țările dezvoltate, s-a schimbat rapid și fundamental ca structură și activități; multe dintre schimbări sînt legate de **noile căi ale generării, stocării, comunicării și utilizării informației**. S-a trecut de la epoca industrială la epoca informațională; aceasta înseamnă că din ce în ce mai mulți oameni manipulează, din ce în ce mai mult, **informația**.

**Calculatoarele și tehnologiile comunicațiilor sînt semnul dramatic și vizibil al revoluției informaționale**. De aceea, alfabetul calculatoarelor a intrat în preocupările multor educatori pentru a adapta școala la cerințele noii epoci. Punîndu-și întrebarea: **„Trebuie copiii să utilizeze calculatoarele în școli?”** mulți educatori au răspuns afirmativ, iar — treptat — copii, părinți și educatori au dorit să învețe despre (și să utilizeze) calculatoare, pentru a se asigura supraviețuirea și bunăstarea într-o lume în stare de schimbare; mulți oameni sînt convinși că **informatica oferă noi șanse pentru copii și pentru profesorii lor**, pentru creativitate și pentru rezolvarea unor probleme mai interesante și mai complexe; alții gîndesc că toți copiii — și adulții — trebuie **„să învețe să controleze mașinile în-vînte ca mașinile să-i ia sub control”**.

România, învățămîntul românesc se angajează intensiv, acum după revoluția din decembrie 89, în acest proces complex, mai încet sau mai rapid, după cîtă vreme va trece pînă la o **conștientizare pe plan național, local și individual** de necesitatea alocării unor resurse și a adoptării unor strategii adecvate recuperării marelui decalaj față de informatizarea educației în lume.

Apariția **prezentei lucrări** constituie un **crez**, atît explicit și implicit, al autorilor și editorilor. Aici nu ne vom mai referi decît la unele **acțiuni și propuneri 1990** ale unora dintre realizatorii cărții cît și la unele **programe și activități** ale Ministerului Învățămîntului.

Stadiul avansat al imprimării lucrării în decembrie 1989 a impus coordonatorului general și redactorului cărții, încă la începutul lunii Ianuarie 1990, discutarea la conducerea Ministerului Învățămîntului a unor probleme legate de apariția iminentă a lucrării și de corelarea cu opțiuni angajate și previzibile ale ministerului respectiv. În acest context, Dl. Ministru adjunct pentru Învățămîntul Preuniversitar, Octavian Stănășilă, și Dl. Ministru al Învățămîntului, Mihai Șora, au solicitat redactorului — în cursul uneia dintre discuții — pregătirea unei evaluări preliminare a problematicei informatizării învățămîntului (din punctul de vedere al unor specialiști în cercetare-proiectare-produție de calculatoare persoane, a unor conceptori de software educațional, al editorilor de profil ș.a.), care să fie prezentată informativ factorilor de decizie din învățămînt. Redactorul cărții a realizat această informare folosind contacte cu specialiștii, cercetări proprii, documentări la sediile UNESCO (inclusiv o discuție consistentă cu D-na Karin Berg, Directoarea CEPES — Unesco din București, din care s-a sintetizat interesul CEPES — Unesco pentru sprijinirea materială și conceptuală a strategiilor de informatizare a învățămîntului românesc). În 3 februarie 1990, cadrul acestei informări a fost expus (la o reuniune organizată de Dl. Ministru Octavian Stănășilă) odată cu propunerea unei mișcări denumite **„Educație și Informatică”**, cu un **nucleu de inițiativă** cuprinzînd atît membri ai colectivului de elaborare a cărții (Prof. dr. ing. Adrian Petrescu, șef catedră dr. ing. Nicolae Țăpuș) cît și alți specialiști, de la ICI, ITC, Fabrica Electronica, Întreprinderea de Calculatoare Electronice cum și din școli generale, licee, învățămînt superior, invitați la inițiativa și prin intermediul redactorului cărții. Ulterior, reprezentanți ai cărții și ai acestui nucleu „Educație și Informatică” au mai prezentat **puncte de vedere** în alte întîlniri din minister, la nivelul inspectoratului de specialitate, al comisiei de informatică nou reînființată în minister, al conducerii ministerului.

Menționăm cîteva:

— Consolidarea mișcării **„Educație și Informatică”** și a nucleului de inițiativă cu factori din **învățămînt, cercetare, producție, comunicații**; atacarea problemelor prioritare prin **analiză sistemică** pentru diferite **nivele de educație și 3 orizonturi de timp**: 1<sup>o</sup>, urgent/imediat, 2<sup>o</sup>, perspectiva 1990—1991 și 1991—2000, 3<sup>o</sup>, persp. ulterioară, de către **echipe** care să propună soluții eficiente pentru informatizarea învățămîntului; în orizontul 1<sup>o</sup> s-a propus evaluarea resurselor informatice, umane și materiale, actuale; planurile pentru concursuri și pentru perioada vacanțelor etc.; în 2<sup>o</sup> — programe analitice, pregătirea profesoriilor, programul de resurse hard și soft, toate analizate de echipe multidisciplinare care să concretizeze un **program național preliminar de educație informatizată**; în 3<sup>o</sup> **plan național pe cîțiva ani, de informatizare a învățămîntului**.

Printre temele susceptibile de analiză: identificarea stadiilor actuale din învățămînt / educație; oferta de echipamente a producției pe clase și în timp;



oferta de programe educaționale, toate realizate prin baterii de chestionare, dirijate în teritoriu, prelucrate pe calculator, cu sinteza rezultatelor; evaluarea pregătirii profesorilor de diferite grade: evaluarea produselor-program din punct de vedere educațional; evaluarea unor structuri de învățământ informatizat; evaluarea concursurilor olimpice; evaluarea cursurilor extrașcolare în cercuri și cluburi; stabilirea unor programe analitice.

Ministerul Învățământului a realizat pînă în aprilie 1990 (prin personalul propriu) o evaluare teritorială și de tipuri de școli, printr-o anchetă care mai continuă; în școli există cîteva mii de calculatoare, îndeosebi HC-85 și compatibile, dar există mari discrepanțe între județe și între școlile din localitățile județului; s-au evaluat, specialitățile și perfecționarea profesorilor, numărul elevilor etc.

— a organizat la Brașov, în aprilie, olimpiada de informatică 1990 (Editura Tehnică va publica problemele propuse și rezolvate, cum a făcut cu cele similare din 1987—89, ce sînt în carte;

— a organizat participarea unor grupe de tineri programatori școlari la o mare întîlnire de profil de Georgia — U.R.S.S.;

— a preluat și repus în funcțiune în cercuri de petrecere a timpului liber, vechile case ale CNOP, U.T.C.;

— a stabilit **Planurile de învățămînt pentru toate stagiile învățămîntului preuniversitar**, din care remarcăm că, de ex., la **învățămîntul gimnazial** în cadrul **pregătirii opționale pentru noțiuni de informatică și tehnică de calcul** au fost stabilite pentru clasele VI — 108 ore; VII — 108 ore; VIII — 102 ore, din care **pentru informatică** aproximativ 1/3 și **pentru tehnică de calcul** — 2/3.

Pentru învățămîntul liceal, în profilul real matematică-fizică în cadrul obiectivului „Pregătire opțională” se prevede pentru informatică și tehnică de calcul cîte 72 ore în clasele IX și X (50% și 50%), cîte 108 ore în clasa XI-a ( $2/3+1/3$ ) și 136 ore în clasa XII-a ( $1/2+1/2$ ).

— Celelalte profile liceale (fizică, umaniste, limbi străine) nu au opțional informatică și tehnică de calcul (fapt oarecum discutabil).

— În schimb, la licee de profil informatic, unde cursurile nu mai sînt facultative, avem situația pe clase, materii, ore:

Cursul/Clasa	IX-a	X-a	XI-a	XII-a	Total ore
— Curs de informatică	144	144	144	136	564
— Laborator	180	144	144	136	604

După cum se vede, o activitate fructuoasă, o recunoaștere a informaticii și tehnicii de calcul, chiar în acest timp scurt de 100 zile.

Menționăm că la 30 aprilie 1990, Dl. Federico Major, Directorul general al UNESCO, aflat la București, a ținut în Aula Bibliotecii Centrale Universitare (BCU) o conferință în care a făcut și un apel la toate guvernele, toate organizațiile și persoanele responsabile din toate țările să ajute la refacerea BCU (distrușă în Revoluția din decembrie 1990) și la informatizarea funcționării sale; s-au făcut referiri la informatizarea învățămîntului românesc, inclusiv cu ajutorul CEPES — UNESCO.

— Pentru luna mai '90 Dl. Ministru adj. Stănășiță cere reactivarea comisiei de informatică, inclusiv prezența membrilor grupului „Educație și Informatică” pentru a ataca frontal informatizarea învățămîntului gimnazial, liceal etc. românesc, folosind concepții superioare, utilizînd atît mijloace cît și experiența străină.

Delegații ale conducerii Ministerului au avut contacte — care vor continua — cu oficialități și firme străine (italiene, germane, engleze, franceze); acestea implică stabilirea unei concepții care a învățămîntului și educației românești, pentru alegerea celor mai adecvate căi de informatizare.

În ce privește ofertele de echipamente și programe românești pentru informatizarea învățămîntului menționăm că nucleul „Educație și Informatică” vede în, linii mari, **continuarea un timp scurt a informaticii pe 8 biți (tip HC-85) și pregătirea unei informaticii pe 16 biți (tip Felix-PC)**; dar neapărat cu sisteme educaționale de tip rețea locală.

Mai menționăm că Dl. Ministru Otto Stamp a acceptat organizarea — în mai 1990 — a unor demonstrații de rețele educaționale pentru școli, organizate de acest nucleu, cu colaborarea ICI și a Fabricii Electronica.

Prin cooperarea mai multor factori interesați (inclusiv a Institutului de Pedagogie (care deja a avut o prezență activă după revoluție) se poate ajunge la selectarea unei gândiri raționale de informatizare, pentru care în carte sînt date o serie de opțiuni internaționale și din experiența din țară.

\* Dintre realizatorii cărții sînt membri ai Comisiei de informatică a Ministerului Învățămîntului, — N. Tăpuș, vicepreședinte și P. Zamfirescu, membru.



## CONTINUARE DIN VOLUMUL 1 DE LA PROLOG—DIALOG—EPILOG

### Elemente privind funcționarea și componentele microsistemului

#### HC-88.

Lansarea sistemului de operare are loc ca urmare a aplicării tensiunii de alimentare sau prin activarea butonului RESET, în cazul în care calculatorul este deja sub tensiune. Dacă în unitatea de disc flexibil A se afla o dischetă, atunci se execută programul încărcător pentru sistemul rezident pe disc: CP/M sau BASIC. Din acest moment sistemul este pregătit pentru a primi comenzile date de utilizator.

Programul de test. Pentru verificarea resurselor hardware ale sistemului, după încărcarea sistemului de operare CP/M, de pe discheta din unitatea A, fapt semnalat de mesajul:

58k CP/M ver.2.2/2                      A>

se tastează numele fișierului cu programul de test: PTHC88 (CR).

Pe ecran va apărea mesajul:            TEST HARDWARE HC88            TEST=

Dupa semnul " = " se așteaptă activarea unei taste care va indica unul din următoarele teste:

- P - test pentru microprocesor ( 280 ),
- E - test pentru ecran ( testeaza memoria video si introduce mira culorilor pentru reglajul monitoarelor color ),
- K - test pentru tastatură ( se verifica funcționarea fiecărei taste ).
- S - test pentru linia serială ( ,test-imprimanta ),
- R - test pentru rețea ( se verifică interfața cu rețeaua ),
- M - test memorie suplimentară ( testele: adresare, BARBER, PING-PONG, WALK ).

Testul de memorie va fi întotdeauna ultimul, deoarece distruge,



prin scrierile pe care le face in memorie, atât informațiile sistemului CP/M, cât și ale programului de test. Dacă se dorește lansarea în secvența de mai sus a tuturor testelor, la început, după " = ", se va activa tasta ( CR ).

#### Copierea Dischetelor:

1. Se introduce discheta sistem in unitatea A si discheta nouă in unitatea B.
2. Daca noua dischetă nu este formatată, se va proceda la formatarea ei prin comanda FORMAT.
3. Se introduce comanda:

DIP B:=\*.\*[V](CR)

In timpul operației de copiere, pe ecran apar mesaje referitoare la fișierele copiate. La sfârșit apare mesajul COPY COMPLETE și prompterul: A>.

Tastatura. Aceasta este constituită din 86 de taste organizate in trei grupuri: normale ( corp alb ), taste folosite in modul extins ( corp gri ) și taste funcționale ( corp rosu ). Tastele sint multifuncționale, semnificațiile lor depinzind de modul de operare al sistemului BASIC ( K, L, EXTINS ). Semnificațiile și modurile de operare ale tastaturii sint descrise pe larg in capitolul de prezentare a limbajului BASIC.

In modul de lucru CP/M tastele cu simboluri compuse ( nume de funcții ) nu sint folosite.

#### Modul de afișare.

In modul de lucru-BASIC, ecranul este împărțit in două zone:

- zona superioară de 22 linii este folosită pentru listarea in-

strucțiunilor sau rezultatelor programului;

- zonă inferioară, de la baza ecranului, este folosită pentru introducerea de comenzi, a liniilor de program, a datelor de intrare, cit și pentru mesaje.

În modul de lucru CP/M ecranul este constituit dintr-o singură zonă cu 24 linii și 80 coloane. Numărul caracterelor tipăribile: 60; numărul caracterelor semigrafice: 16; celula de caracter: 6x8 puncte; matricea de caracter: 5x7 puncte; culori, cerneală, hirtie, margine/bordură: 8.

#### Realizarea CP/M HC-88.

Varianta CP/M pentru microcalculatorul HC-88 a fost realizată prin procedeul uzual de rescriere a modulului software BIOS. Componentele BDOS și CCP au rămas neschimbate, astfel încât funcțiile sistemului de operare, văzute de programator, au rămas nemodificate.

În continuare se mai menționează particularități ale actualei implementări pentru sistemul de operare HC-88.

#### Copierea discului sistem.

După încărcarea sistemului de operare și apariția mesajului arătat mai sus, se poate face o copiere a discului sistem. Se introduce discul sistem original în unitatea A și se lansează sistemul de operare. Se lansează utilitarul FORMAT. La întrebarea :

"Drive to use (A/B)" se alege B. La întrebarea "Format 1=BASIC 2=CP/M (1/2)" se alege deocamdata opțiunea 1, adică format BASIC. Se introduce discul nou în unitatea B și se tastează RETURN. După terminarea operației apare mesajul "Format complete". Se răspunde cu N, la întrebarea: "Format another ( Y/N )"



Se copiază fișierul CPM88 de pe discul original pe discul copie.

Se folosește comanda DIP: DIR B:=CPM88[V].

Este important ca primul fișier copiat pe noul disc să fie CPM88, deoarece poziția acestui fișier care conține sistemul de operare este fixă. Fișierul CPM88 trebuie să se afle în primele grupuri de alocare după directorul discului. Discul copie poate fi apoi completat cu programele tranzitorii: fișierele .com și cu alte fișiere.

### Transferuri CP/M - BASIC și invers.

Formatul discului CP/M este identic cu formatul folosit de HC-88 cu disc cu dublă densitate și de HC-88 în modul BASIC. În acest fel se poate face un transfer simplu între cele două moduri de lucru ale calculatorului HC-88: în modul CP/M este recomandabil să se introducă discul BASIC în unitatea B, deși el poate fi citit/scriș foarte bine și în unitatea A.

De exemplu, se introduce în unitatea A un disc CP/M HC-88, iar în unitatea B un disc BASIC HC-88. Se tastează comanda DIR; pe ecran apare conținutul discului BASIC în formatul specific CP/M, dar altfel identic cu conținutul care se obține în modul BASIC la tastarea comenzii CAT. Cu discul BASIC se pot folosi și alte comenzi, cu observația că discul BASIC nu poate fi pus în unitatea A, pentru a încărca la rece sau la cald sistemul CP/M.

BIOS-ul detectează automat tipul discului inserat, la prima operație de scriere sau citire efectuată după o încărcare la rece sau la cald a sistemului de operare CP/M. Astfel, singura regulă care se impune este activarea tastelor care generează CTRL/C, la nivelul prompterului CCP, după schimbarea discurilor.

**CONTINUARE ÎN VOLUMUL 2  
LA PROLOG—DIALOG—EPILOG**

# CUPRINS

## Volumul 2

Cuprins general (vol. 1 și 2) .....	V
PROLOG—DIALOG—EPILOG (continuare din vol. 1) .....	VI
Cuprins detaliat vol. 2 .....	XIV

### Partea a VI-a: Microcalculatorul HC-85 în procese industriale. Pachete de programe de aplicații pentru HC-85 1

<b>Capitolul 13. Proiectarea interfețelor pentru echipamentele nestandard, cu aplicații în măsurători și conducerea microroboților industriali</b> .....	1
13.1. Elemente privind proiectarea interfețelor pentru HC-85 ..	1
13.2. Cuplarea unei interfețe programabile PIO, la HC-85 .....	8
13.3. Determinarea simultană a două constante fizice $\sigma$ și $g$ , în cadrul unui experiment asistat de HC-85 .....	13
13.4. Conducerea minirobotului RIP 0.2 cu HC-85 .....	17
<b>Capitolul 14. Pachete de programe aplicative: Grafica 3-D, Baze de date. Tabelare electronică. Prelucrarea textelor</b> .....	25
14.1. Program pentru grafica în trei dimensiuni .....	25
14.2. Baze de date .....	27
14.3. Tabelare electronică .....	30
14.4. Procesor de texte .....	34

### Partea a VII-a: Calculatorul personal în învățământ și educație 44

<b>Capitolul 15. Calculatorul personal în procesul de învățământ</b> .....	44
15.1. Informatica în programa învățământului liceal .....	44
15.2. Asistarea cu calculatorul a procesului de învățământ .....	48
15.3. Tipuri de laboratoare pentru procesul de învățământ .....	51
15.4. Rezolvarea unei probleme cu calculatorul. Etape; algoritmi și reprezentările lor; exemple ..	59
15.5. Scurtă sinteză privind elaborarea programelor .....	68
15.6. Calculatoare personale în învățământul din lume (sumar) ..	71
<b>Capitolul 16. Educație și informatică</b> .....	83
16.1. Direcții, concluzii și programe ale primului congres internațional UNESCO; Educație și informatică .....	83
16.2. O nouă programă analitică americană pentru „alfabetizarea” în informatică .....	87
16.3. Programul disciplinei „Bazele informaticii și tehnicii de calcul”, învățământ mediu U.R.S.S. ....	92
16.4. Aplicații informatice educaționale în diferite țări (Franța, Japonia) .....	96
16.5. Concursuri de informatică naționale. Probleme rezolvate (continuă în anexe) .....	97
16.6. Teme rezolvate pentru cluburile de informatică (continuă în anexe) .....	101



## Partea a VIII-a: Programe educaționale în BASIC, pe calculatorul HC-85

### Capitolul 17. Programe pentru clasele I—VIII

17.1. Adunarea .....	113	17.16. Boyle — Mariotte (II) .....	125
17.2. Test — adunarea .....	113	17.17. Frecarea .....	126
17.3. Scăderea .....	116	17.18. Concentrația .....	126
17.4. Înmulțirea .....	117	17.19. Numere polindroame (cerc de matematică) .....	127
17.5. Împărțirea .....	117	17.20. Relații .....	128
17.6. C.M.M.D.C. ....	118	17.21. Rot triunghi .....	129
17.7. C.M.M.M.C. ....	118	17.22. Rot. trapez .....	129
17.8. Conversie .....	119	17.23. Rot. hexagon .....	129
17.9. Ec. Gr.I .....	120	17.24. Polinom .....	130
17.10. Sistem $2 \times 2$ .....	121	17.25. Triunghi .....	131
17.11. Funcția $X \rightarrow a \cdot X$ , $X \in R$ .....	122	17.26. Fibonacci .....	132
17.12. Funcția $X \rightarrow X^2 -  X $ , $X \in R$ .....	123	17.27. Trasare loc geometric .....	133
17.13. ABC 2 .....	123	17.28. Desenarea unei table de șah .....	133
17.14. Funcția $X \rightarrow a \cdot X^2$ , $X \in R$ .....	124	17.29. Grafice de funcții (cerc de matematică) .....	134
17.15. Boyle — Mariotte (I) .....	125		

### Capitolul 18. Programe pentru clasele IX—XII

18.1. Funcția modul .....	137	18.20. Trasare elipsa .....	153
18.2. Funcția $f(x) = (x+1)(3 -  x )$ , $x \in R$ .....	139	18.21. Trasare hiperbola .....	153
18.3. Funcția $f(x) =  x^2 - 5x + 6 $ , $x \in R$ .....	139	18.22. Trasare parabola .....	153
18.4. Funcția $f(x) = ( x +1)(3 -  x )$ , $x \in R$ .....	140	18.23. Trasare loc geometric .....	154
18.5. Grafice — funcții sinus, folo- sind modulul .....	140	18.24. Grafice — funcții exponenția- le și logaritmice .....	155
18.6. Grafice — funcții sinus .....	142	18.25. Val. medie și abatere .....	157
18.7. Funcții cosinus .....	143	18.26. Dreptunghi .....	157
18.8. Grafice — funcții cosinus, fo- losind modulul .....	143	18.27. Simpson .....	158
18.9. Arcsin .....	144	18.28. Compunere .....	158
18.10. Arccos .....	144	18.29. Teoria cinetico-moleculară .....	159
18.11. Tangentă .....	145	18.30. Lentila .....	161
18.12. Cotangentă .....	146	18.31. Optică .....	162
18.13. Inecuații și sisteme de inecu- ații (cerc de matematică) .....	148	18.32. Motoare termice .....	163
18.14. Ecuația de gradul II în C .....	150	18.33. Sim. com. ....	165
18.15. Logaritmi .....	151	18.34. Osciloscopul .....	166
18.16. N factorial .....	151	18.35. Imagine — osciloscop .....	166
18.17. Combinări .....	151	18.36. Normalitatea .....	167
18.18. Produs matricial .....	152	18.37. Molaritatea .....	167
18.19. Transpusa .....	151	18.38. Seria reactivității .....	168
		18.39. Ultima cifră .....	171
		18.40. Modulul .....	175
		18.41. TRIGRAF .....	177
		18.42. ACCO .....	177

## Partea a IX-a: Microbiblioteca de programe pe casete

### Capitolul 19. Prezentarea casetelor cu programe pentru calculatorul HC-85

19.1. Sinteză privind conținutul case- telor .....	182	19.3. Caseta II. Prezentarea și încăr- carea programelor .....	188
19.2. Caseta I. Prezentarea și încăr- carea programelor .....	186	19.4. Caseta III. Prezentarea și încăr- carea programelor .....	190



## Partea a X-a: Complemente matematice

192

### Capitolul 20. Sisteme de numerație poziționale

192

20.1. Noțiuni introductive .....	192
20.2. Sistemul de numerație binar ..	194
20.3. Sistemul de numerație octal ..	201

20.4. Sistemul de numerație hexaze- cimal .....	203
--	-----

### Capitolul 21. Operatori logici

206

21.1. Operatori logici de bază ....	206
21.2. Proprietăți ale operatorilor lo- gici .....	111

21.3. Funcții logice .....	210
21.4. Problema deciziei .....	213

### Capitolul 22. Noțiuni de algebră booleană

215

22.1. Scurtă prezentare .....	215
22.2. Expresii boolene .....	219
22.3. Funcții boolene .....	221

22.4. Forme canonice ale funcțiilor boolene .....	226
22.5. Simplificarea funcțiilor boole- ene .....	230

## Partea a XI-a: Complemente informatice

236

### Capitolul 23. Instrucțiunile microprocesorului Z 80

236

### Capitolul 24. Încărcarea și lansarea în BASIC a unor programe scrise în limbaj de asam- blare / cod mașină

241

### Capitolul 25. Calculatorul personal profesional Felix PC (compatibil IBM PC). Noua familie IBM PS/2

259

25.1. Calculatorul personal-profesio- nal Felix PC .....	241
---	-----

25.2. Noua familie de calculatoare IBM-PS/2 .....	251
--	-----

## BIBLIOGRAFIE GENERALĂ

275

ANEXE 1. Metodica reprezentării algoritmilor .....	278
2. Programe educaționale HC-85, la ICI .....	280
3. Interpretorul BETA BASIC .....	283

4. Concursuri informatice naționa- le (probleme rezolvate — continu- are din vol. 1, 16.5) .....	293
5. Probleme în pseudocod .....	302
6. Soluții ale temelor din 16.6 .....	305

### PROLOG — DIALOG — EPILOG (continuare din vol. 2, pag. XIII). HC extins, HC-88 (disc, rețea)

320  
— 360



25.09.1994  
Dumitru

## **MICROCALCULATORUL HC-85 ÎN PROCESE INDUSTRIALE. PACHETE DE PROGRAME DE APLICAȚII PENTRU HC-85.**

### Capitolul 13.

### **Proiectarea interfețelor pentru echipamentele nestandard, cu aplicații în măsurări și conducerea microroboților industriali.**

#### **13.1. Elemente privind proiectarea interfețelor pentru HC-85**

Microcalculatorul HC-85 poate fi utilizat în numeroase aplicații în care el joacă rolul unui echipament pentru achiziții și prelucrări de date, de la un proces sau pentru comanda efectivă a unui proces.

Un proces fizic este caracterizat prin diferite mărimi, care pot fi preluate de la intrările și ieșirile sale. Mărimile respective au naturi fizice diferite: deplasări liniare/unghiulare, viteze, accelerații, debite, presiuni, temperaturi, concentrații, curenți electrici, tensiuni electrice etc.

În cazul în care calculatorul trebuie să achiziționeze datele corespunzând acestor mărimi fizice, este necesar ca ele să fie transformate în semnale electrice (tensiuni, curenți), cu ajutorul unor echipamente, numite traductoare.

Semnalul electric (de exemplu tensiunea) furnizat la ieșirea unui traductor poate avea în timp o variație continuă sau discretă.

În cazul semnalului cu variație continuă, este necesară conversia lui într-un semnal numeric cu variație discretă, în vederea prelucrării numerice. În figura 13.1 se prezintă sub forma cea mai generală un convertor analog/numeric



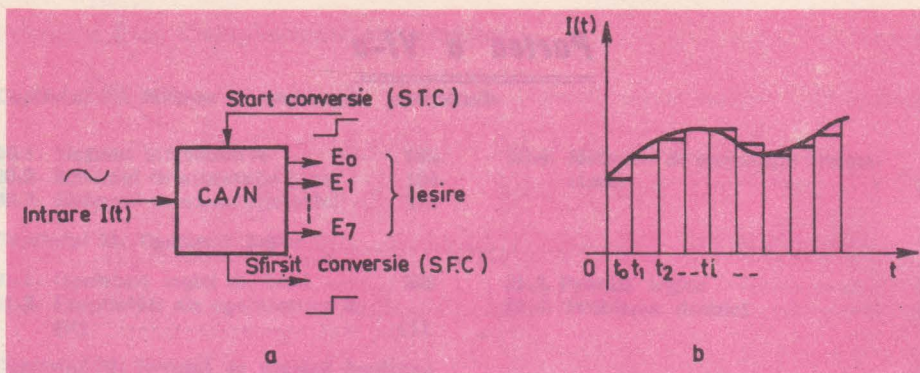


Fig. 13.1. Conversie analog-numerică: a) reprezentarea unui convertor analog-numeric; b) eșantionarea semnalului cu variație continuă  $I(t)$

(CA/N), folosit în acest scop. Semnalul de intrare  $I$ , cu variație continuă, este transformat într-un semnal cu variație discretă, la momentele de timp  $t_0, t_1, \dots, t_i, \dots$ . Ieșirea va fi reprezentată prin semnalele:  $E_7, \dots, E_i, \dots, E_0$ , care pot lua numai două valori:  $E_{\min}$  (circa 0V) și  $E_{\max}$  (circa +5V). Aceste niveluri de tensiune sînt puse în corespondență cu cifrele binare 0 și 1, astfel că mărimea de intrare  $I$  va fi reprezentată, la momentele  $t_i$ , printr-un număr binar cu rangurile  $E_7, \dots, E_0$ . În cazul de față, considerînd că  $I_{\min}$  și  $I_{\max}$  au drept corespondente la ieșire valorile 0 (în binar: 00...0) și respectiv 255 (în binar: 11...1), se constată că rezoluția convertorului A/N este de  $1/256$  din valoarea maximă a mărimii de intrare. Semnalul de start conversie (S.T.C) este dat de calculator, în timp ce semnalul de sfârșit conversie (S.F.C) este furnizat de convertor. După lansarea operației de conversie, calculatorul citește periodic semnalul S.F.C, care indică sfârșitul operației de conversie, cînd trece pe nivel ridicat. În continuare, datele  $E_7, \dots, E_0$  sînt citite de calculator, de la portul de intrare asociat datelor convertorului.

Calculatorul poate furniza la intrarea unui convertor numeric/analogic (CN/A) secvențe de numere binare, care sînt transformate într-un semnal cu variație discretă, ce se poate colecta la ieșirea convertorului (fig. 13.2).

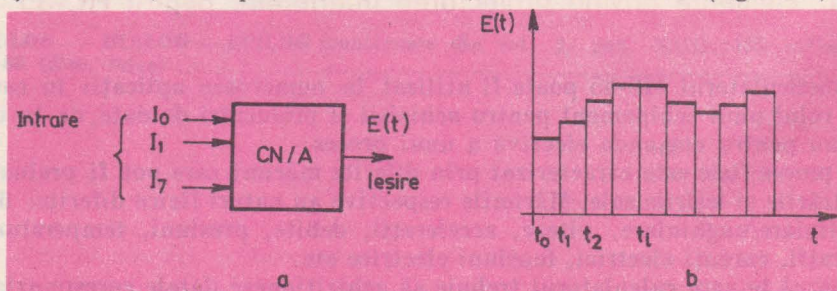


Fig. 13.2. Conversia numeric-analogică: a) reprezentarea unui convertor numeric-analogic; b) semnalul la ieșirea convertorului numeric-analogic.

În acest mod, din proces se pot colecta semnale cu variație continuă, se convertesc în semnale numerice, se prelucrează și rezultatele sînt furnizate procesului sub forma de semnale cu variație discretă. Dacă operațiile de conversie și prelucrare se desfășoară într-un interval scurt de timp, în comparație cu perioadele semnalelor ce caracterizează procesul, se poate afirma că ele au loc în timp real.

**VI. HC-85 ÎN PROCESE. PACHETE PROGRAME**



Adesea procesele fizice sînt descrise și prin mărimi care iau numai două valori discrete, ce pot fi puse în corespondență cu elementele mulțimii binare [0, 1]. Asemenea semnale caracterizează stările unor comutatoare bipoziționale, ale unor contacte de relee folosite în acționări, ale unor tranzistoare blo-

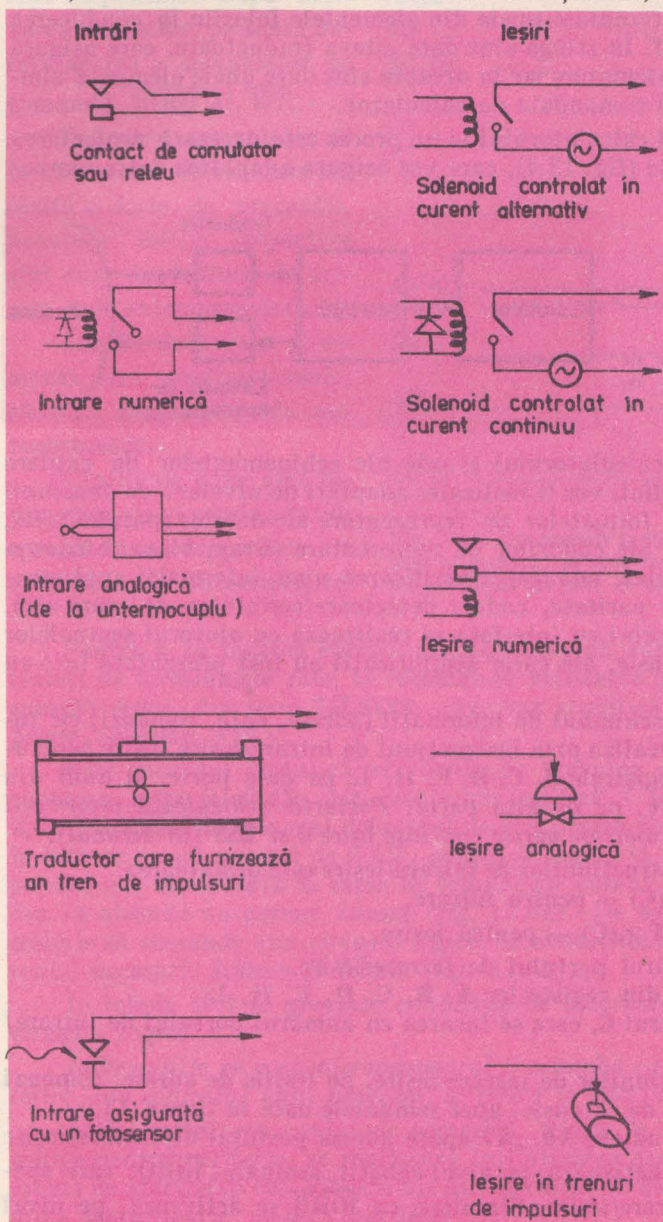


Fig. 13.3. Unele traducătoare și elemente de execuție folosite în conducerea cu calculatorul a proceselor industriale

cate sau în conducție etc. Pe baza unor astfel de semnale numerice, calculatorul poate lua decizii privind intervenția în proces, ceea ce se realizează prin generarea unor semnale de ieșire binare corespunzătoare.

### 13. INTERFEȚE PENTRU MĂSURĂRI ȘI ROBOȚI



În rezumat, calculatorul, prevăzut cu circuitele de cuplare necesare, poate achiziționa date reprezentând semnale continuu variabile (analogice) și semnale discrete-numerice (binare). De asemenea, informația prelucrată poate fi furnizată de către proces sub forma unor semnale analogice și/sau numerice.

În figura 13.3 sînt prezentate unele din elementele folosite în conducerea proceselor cu calculatorul. În stînga sînt date cîteva traductoare, care asigură semnalele de intrare în calculator, iar în dreapta sînt date unele elemente simple de execuție, ce pot fi comandate de calculator.

În vederea conectării calculatorului la un proces este necesară proiectarea și realizarea unor interfețe (fig. 13.4), care vor asigura adaptarea între semna-

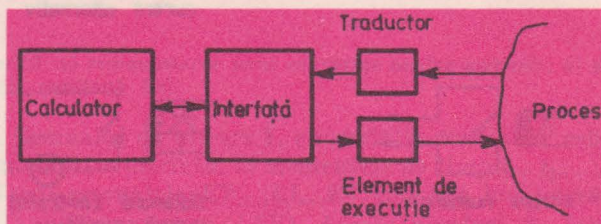


Fig. 13.4. Schema generală de cuplare a unui calculator de proces

lele electrice specifice calculatorului și cele ale echipamentelor de cuplare nemijlocită la proces. Astfel, vor fi realizate: adaptări de niveluri de tensiuni/curenți, modificări ale formatelor de reprezentare ale datelor (paralel/serie, serie/paralel), schimbări ale codurilor de reprezentare (Gray, binar, binar-zecimal, ASCII, EBCDIC etc), adăugarea/verificarea unor informații suplimentare de control (biți de paritate, coduri detectoare/corectoare de erori etc). Dialogul microcalculatorului cu interfața se realizează cu ajutorul semnalelor de la conectorul de extensie, ale căror semnificații au fost prezentate într-un capitol anterior.

În cele ce urmează, schimbul de informații (adrese, date, comenzi) cu interfața se consideră a se realiza prin instrucțiuni de intrare/ieșire, între acumulatorul A sau unul din registrele B, C, D, E, H, L, pe de-o parte, și unul din porturile de intrare/ieșire, pe de altă parte. *Porturile vehiculează informații structurate pe 8 biți și au asociate adrese cuprinse între 0 și 255 (în zecimal).*

Forma generală a instrucțiunilor de intrare/ieșire este următoarea:

**IN A,(nn)** sau **IN r,(C)** — pentru intrare,

**OUT A,(nn)** sau **OUT r,(C)** — pentru ieșire,

unde: — **nn** este numărul portului de intrare/ieșire,

— **r** este unul din registrele: A, B, C, D, E, H, L,

— **C** este registrul C, care se încarcă cu numărul portului de intrare/ieșire.

La execuția instrucțiunilor de intrare-ieșire, pe liniile de adrese, comenzi și date, ale conectorului de extensie, apar semnalele date în figura 13.5.

Astfel, pe liniile de adrese A0,...,A7 apare adresa portului de intrare/ieșire manipulat, după care se activează, pe nivel coborît, semnalul  $\overline{\text{IORQ}}$ , care specifică o operație de intrare/ieșire. Simultan cu  $\overline{\text{IORQ}}$  se activează, pe nivel coborît, unul din semnalele  $\overline{\text{RD}}$  (intrare) sau  $\overline{\text{WR}}$  (ieșire).

Pentru intrare, datele trebuie să fie forțate de sursă, pe liniile D0,...,D7, spre sfîrșitul intervalului de timp, cînd  $\overline{\text{IORQ}}$  și  $\overline{\text{RD}}$  sînt active.



În cazul ieșirii, datele furnizate de microprocesor sînt disponibile după activarea adreselor, pe toată durata activă a semnalelor de comandă  $\overline{IORQ}$  și  $\overline{WR}$ .

Folosirea semnalelor de la conectorul de extensie necesită o serie de precauții, deoarece liniile respective sînt conectate direct la terminalele microprocesorului.

Liniile de adrese și comenzi sînt unidirecționale; ele sînt comandate de microprocesor.

Liniile de date sînt bidirecționale și conduc datele spre/de la microprocesor, în operațiile de intrare/ieșire.

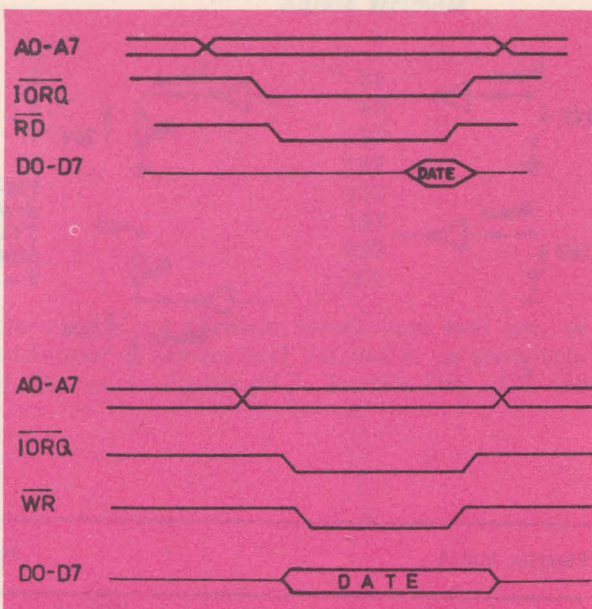


Fig. 13.5. Diagramele temporare ale semnalelor de adresă, date și comenzi pentru operațiile de citire (a) și scriere (b) de la/la un post de intrare/ieșire

Pentru a nu încărca excesiv microprocesorul, sub aspectul curenților controlați de terminalele sale, se recomandă utilizarea unor circuite integrate cu consum redus, din seria 74LSxxx, unde xxx specifică funcția circuitului (*Low power Schottky TTL*).

Este cunoscut faptul că nivelurile de tensiune care codifică 0 logic sînt cuprinse între 0—0,4V, iar cele care codifică 1 logic pot lua valori între 2,5—5V. Cazurile cele mai defavorabile sînt acelea cînd circuitul este controlat la intrare cu un semnal de 0,4V, pentru 0 logic, și 2,4V, pentru 1 logic. În primul caz el va forța în sursa de semnal un curent  $I_{i0max}$ , iar în al doilea caz va absorbi un curent  $I_{ilmax}$  (fig. 13.6a). De asemenea, un circuit logic trebuie să comande alte circuite logice, în condițiile de mai sus, asigurînd curenții necesari:  $I_{e0min}$  și  $I_{elmin}$  (fig. 13.6b).

În tabela 13.1 se prezintă condițiile de încărcare pentru cîteva tipuri de circuite logice.

*Informațiile pe care microprocesorul le schimbă cu un echipament de intrare/ieșire, sub controlul unui program, se referă la date, comenzi și stări.*

Astfel, printr-un cuvînt de comandă, trimis de către microprocesor, spre echipamentul periferic, se realizează inițializarea acestuia, se dau diverse comenzi de start/stop etc. Fiecare bit din cuvîntul de comandă poate avea o anumită semnificație, pentru periferic.

Echipamentul periferic se caracterizează prin diverse condiții: alimentat/nealimentat, motor pornit/oprit, echipament ocupat/liber, apariția/absența unei erori etc. Aceste situații au drept corespondent anumiți biți din cuvîntul de stare al echipamentului.

### 13. INTERFEȚE PENTRU MASURĂRI ȘI ROBOȚI



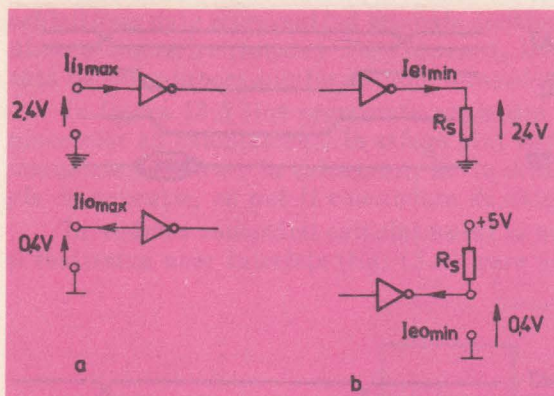


Fig. 13.6. Curenții la intrarea (a) și ieșirea (b) ale unui circuit logic inversor TTL, în condițiile cele mai defavorabile, pentru reprezentarea lui 1 — logic și 0 — logic

Tabela 13.1

Familia logică	74LSxx <sup>1</sup>	74xx <sup>2</sup>	74Sxx <sup>3</sup>
Posibilități de comandă la ieșire			
1 Curentul minim ( $I_{e1min}$ ) garantat ca sursă cu ieșirea la 2,4 V	400 $\mu$ A	400 $\mu$ A	1 mA
0 Curentul minim ( $I_{e0min}$ ) absorbit cu ieșirea forțată la 0,4 V	8 mA	16 mA	20 mA
Posibilități de încărcare a intrărilor			
1 Curentul maxim ( $I_{i1max}$ ) absorbit, cu intrarea la 2,4 V	20 $\mu$ A	40 $\mu$ A	50 $\mu$ A
0 Curentul maxim ( $I_{e0max}$ ) furnizat, cu intrarea la 0,4 V	0,36 mA	1,6 mA	2 mA

<sup>1</sup> TTL Low Power Schottky; <sup>2</sup> TTL obișnuit; <sup>3</sup> TTL Schottky.

Astfel, prin program, unitatea centrală poate citi starea echipamentului, îl poate comanda și, în anumite condiții, poate efectua schimbul de date.

Aceste informații sînt vehiculate prin porturi de intrare/ieșire, plasate în spațiul de adresare disponibil pentru instrucțiunile de intrare/ieșire ale microcalculatorului HC-85. Selecția porturilor se realizează prin adresele specificate de instrucțiunile de intrare/ieșire.

Microcalculatorul HC-85, în configurațiile standard sau extinsă, nu folosește biții de adrese A7, A6, A5, ceea ce permite, prin stabilirea unor valori fixe, pentru biții: A4, A3, A2, A1, A0=1111, selectarea a 8 porturi de intrare și a 8 porturi de ieșire. Astfel, se vor putea utiliza următoarele adrese

## VI. HC-85 ÎN PROCESE, PACHETE PROGRAME



Adrese binare								Adrese zecimale	
A7	A6	A5	A4	A3	A2	A1	A0		
0	0	0	1	1	1	1	1	31	
0	0	1	1	1	1	1	1	63	
0	1	0	1	1	1	1	1	95	
0	1	1	1	1	1	1	1	127	
1	0	0	1	1	1	1	1	159	
1	0	1	1	1	1	1	1	191	
1	1	0	1	1	1	1	1	223	
1	1	1	1	1	1	1	1	255	

Selecția porturilor de intrare/ieșire, cu adresele menționate mai sus, se poate realiza cu circuitul decodificator binar-zecimal 74LS42 și cu circuitele logice auxiliare 74LS10 și 74LS32, ca în figura 13.7. Ieșirile **D0**,...,**D7** vor

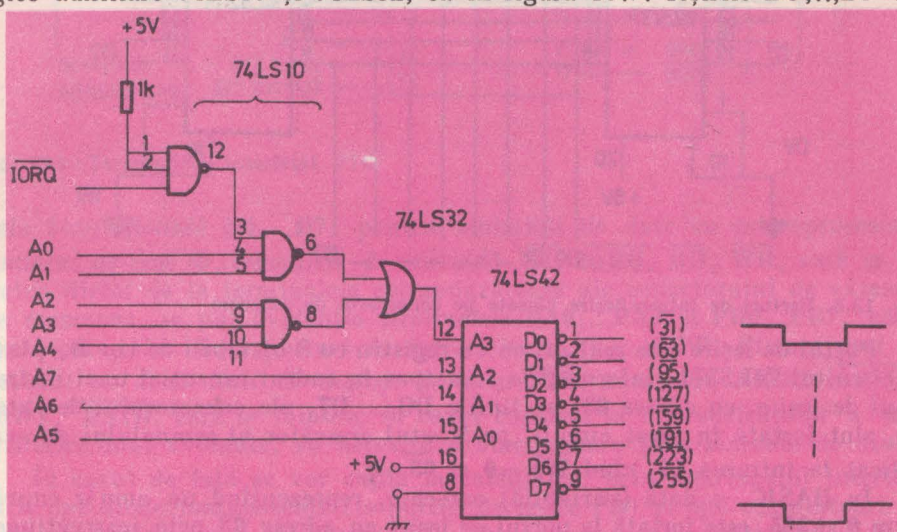


Fig. 13.7. Decodificarea adreselor A0—A7

fi active, pe nivel coborât, succesiv, pentru fiecare din cele 8 adrese aplicate la liniile **A0**,...,**A7**. Semnalele active pe nivel coborât, obținute la ieșirile **1**,...,**9**, corespund activării adreselor **31**, **63**,...,**255**.

În figura 13.8 sunt prezentate două porturi: unul de intrare și altul de ieșire, cu adresa **63**.

Portul de intrare este realizat cu circuitul 74LS245, care joacă rolul unui comutator între liniile de intrare **D0**,...,**D7** și liniile de date **D0**,...,**D7**, ale conectorului de extensie. Circuitul este activat prin semnalul  $\overline{CE}$ , activ pe nivel coborât, când se execută o instrucțiune de intrare cu adresa **63** ( $\overline{RD}=0$  și  $\overline{63}=0$ )

În BASIC variabila **X** va lua valoarea plasată la portul de intrare cu adresa **63**, când se execută instrucțiunea:

**LET X=IN 63**

Când circuitul 74LS245 este dezactivat ( $\overline{CE}=1$ ), ieșirile sale prezintă o impedanță mare, astfel încât liniile corespunzătoare ale conectorului de extensie nu sunt afectate (încărcate).

### 13. INTERFEȚE PENTRU MĂSURĂRI ȘI ROBOȚI



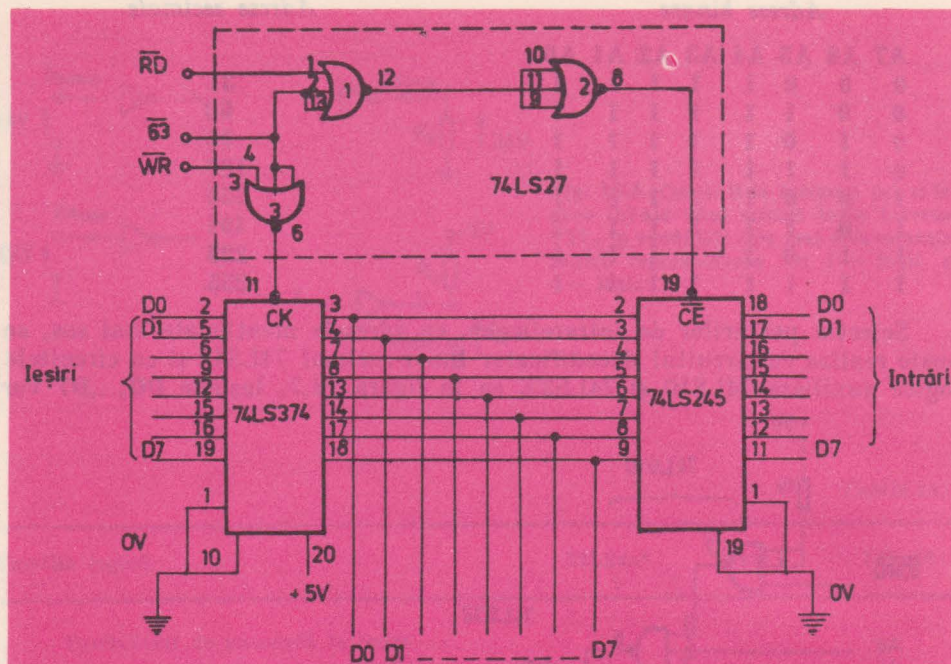


Fig. 13.8. Porturi de intrare/ieșire plasate pe conectorul de extensie

Portul de ieșire este realizat cu un registru cu 8 bistabili de tip **D**, plasat în circuitul 74LS374. Informațiile care apar în cadrul execuției unei instrucțiuni de ieșire, cu adresa **63**, pe liniile **D0**,...,**D7**, ale conectorului de extensie, sînt forțate în acest circuit, pe frontul crescător al semnalului de ceas, aplicat la intrarea **11**, cînd  $\overline{WR}=0$  și  $\overline{63}=0$ .

În BASIC, o dată (variabilă) oarecare, reprezentînd un număr cuprins între **0** și **255**, este forțată la portul de ieșire cu adresa **63** prin instrucțiunea:

**OUT 63, data**

Data va rămîne stocată în bistabilii portului de ieșire, pînă la modificarea ei de către o altă instrucțiune.

### 13.2. Cuplarea unei interfețe programabile PIO la microcalculatorul HC-85\*

Circuitul PIO a fost proiectat în mod special pentru a fi conectat cu microprocesorul Z80, în scopul implementării unei interfețe paralele programabile. Examinînd terminalele acestui circuit (fig. 13.9) se pot evidenția: *magis-*

\* Lucrare prezentată de elevul Petrescu Iacob, clasa a XII-a, Liceul Dimitrie Cantemir, la Sesiunea de comunicări științifice și referate, ale elevilor din Municipiul București, mai 1987.



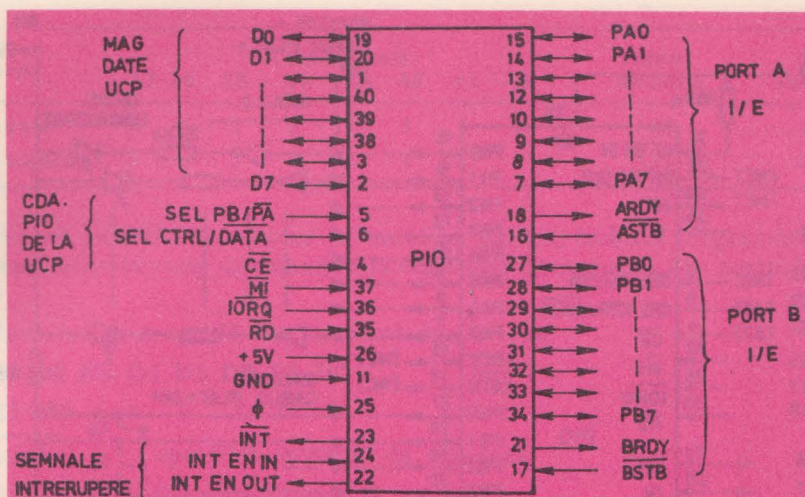


Fig. 13.9. Terminalele interfeței PIO

trala bidirecțională  $D0, \dots, D7$ , pentru schimbul de date cu microprocesorul, semnalul de ceas  $\Phi$ , semnalele de comandă  $\overline{M1}$ ,  $\overline{IORQ}$ ,  $\overline{RD}$ ,  $\overline{WR}$ , care se pot prelua direct de la terminalele corespunzătoare ale conectorului de extensie. De asemenea, se pot evidenția terminalele  $PA0, \dots, PA7$  și  $PB0, \dots, PB7$ , corespunzătoare celor două porturi de intrare/ieșire, PA și PB. Fiecare port este prevăzut cu două linii de dialog:  $\overline{RDY}$  și  $\overline{STB}$ , notate cu  $\overline{ARDY}$ ,  $\overline{ASTB}$  și  $\overline{BRDY}$ ,  $\overline{BSTB}$ , care se folosesc în cazul cuplării la microcalculator a unor periferice mai evolute, ce necesită o sincronizare prin semnale de dialog.

În cazul de față se vor cupla la calculator, prin intermediul interfeței PIO, opt diode luminescente (la portul de ieșire PA) și opt comutatoare bi-pозиționale (la portul de intrare PB). De asemenea, pe biții  $PA0$  și  $PA1$ , ai portului de ieșire PA, se va conecta o schemă de comandă a sensului de rotație al unui micromotor de curent continuu. Aceste periferice simple nu necesită semnale de dialog.

Întrucât circuitele de ieșire ale portului PA nu pot comanda curenți mai mari decât o sarcină standard TTL, între terminalele portului PA și diodele luminescente s-au plasat circuite inversoare, cu colectorul în gol, 7404, care pot comanda 10 sarcini TTL.

În figura 13.10 se prezintă schema interfeței, din care rezultă utilizarea a două circuite integrate 7404 și a unor cabluri plasate pentru efectuarea conexiunilor între circuitul imprimat, care conține blocul diodelor luminescente, (cu rezistențele de limitare) și blocul comutatoarelor, pe de-o parte, și placheta pe care se află interfața PIO, pe de altă parte.

Blocurile diodelor luminescente și comutatoarelor se pot alimenta de la sursa de +5V a calculatorului, consumul total fiind mai mic de 15 mA.

După cum se cunoaște interfața PIO posedă două, registre de date și două registre de comenzi, asociate porturilor PA și PB.

### 13. INTERFEȚE PENTRU MASURĂRI ȘI ROBOȚI



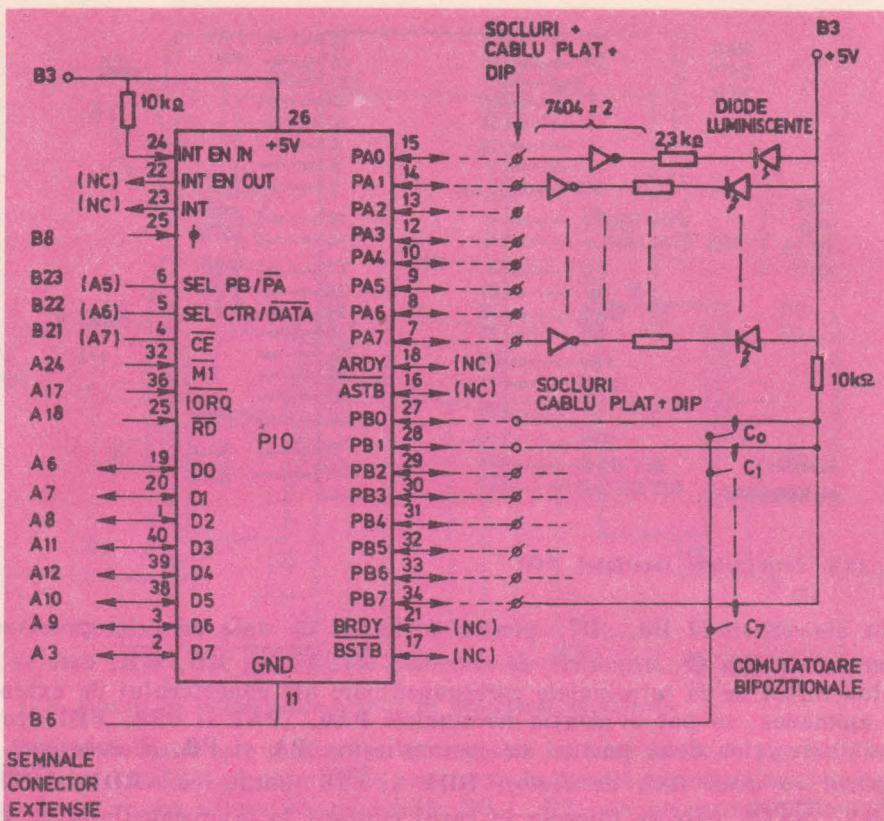


Fig. 13.10. Cuplarea interfeței PIO la conectorul de extensie, la diodele luminescente și la comutatoare

Pentru a înregistra date și comenzi sau pentru a prelua date de la registre, se impune selecția lor, cu ajutorul unor adrese amplasate în spațiul de adresare al porturilor de intrare/ieșire pentru microprocesorul Z80.

Calculatorul HC-85 nu folosește liniile de adrese A7, A6 și A5, pentru selecția echipamentelor periferice standard. Astfel, liniile de adrese A7, A6, A5 vor fi utilizate după cum urmează:

- A7, pe nivel coborât, va selecta PIO, jucând rolul semnalului  $\overline{CE}$ ;
- A6 va specifica selecția unui registru de comenzi (pe nivel ridicat) sau a unui registru de date (pe nivel coborât).
- A5 va asigura selecția portului PB (pe nivel ridicat) sau a portului PA (pe nivel coborât).

Liniile de adrese A4, ..., A0 vor fi menținute la nivel ridicat, pentru a nu interfereze cu alte adrese de porturi de intrare/ieșire, folosite de HC-85.

Selecție	Adresă binară								Adresă zecimală	Nume simbolic
	A7, A6,	A5,	A4,	A3,	A2,	A1,	A0			
MA — date	0	0	0	1	1	1	1	1	31	DA
PA — comenzi	0	1	0	1	1	1	1	1	95	CA



Selecția	Adresă binară								Adresă zecimală	Nume simbolic
	A7	A6	A5	A4	A3	A2	A1	A0		
PB — date	0	0	1	1	1	1	1	1	63	DB
PB — comenzi	0	1	1	1	1	1	1	1	127	CB

Pentru ca PIO să opereze în maniera dorită, trebuie programată, prin înscrisirea unor cuvinte de comandă în registrele de comandă ale porturilor PA și PB.

*Cuvîntul de comandă are structura de mai jos:*

**D7 D6 D5 D4 D3 D2 D1 D0**

**M1 M0 \* \* 1 1 1 1**

**mod** specifică cuvîntul de comandă

\* — biți indiferenți (0 sau 1).

*Cîmpul mod are următoarele semnificații:*

**M1 M0**

0 0 — mod 0 — ieșire pe octet, cu dialog,

0 1 — mod 1 — intrare pe octet, cu dialog,

1 0 — mod 2 — intrare/ieșire pe octet cu dialog,

1 1 — mod 3 — comandă.

*Deoarece se va folosi modul comandă, se va selecta modul 3, cu un cuvînt de comandă: 11111111=255 — în zecimal.*

Pentru a programa PA ca port de ieșire, se va trimite la adresa CA cuvîntul de comandă 255, folosind instrucțiunea:

**OUT CA, 255**

În continuare trebuie să se trimită, tot la adresa CA, un cuvînt care va specifica, la nivel de bit, faptul că este vorba de un bit de ieșire (0) sau un bit de intrare (1). În cazul de față toți biții portului PA sînt programați ca ieșiri. Al doilea cuvînt de comandă va fi: 00000000=0 — în zecimal. El va fi forțat în portul CA cu instrucțiunea:

**OUT CA, 0**

La adresa DA se pot trimite, în continuare, configurații de biți care urmează a se afișa la diodele luminescente.

În cele ce urmează se vor da cîteva exemple de programare a interfeței PIO.

Program pentru afișarea la portul PA, a numerelor binare cuprinse între 0 și 255:

10 LET DA=31: LET CA=95

20 OUT CA, 255: OUT CA, 0

30 OUT DA, 0

100 FOR n=0 TO 255

110 OUT DA, n

120 NEXT n

130 GO TO 100



Programarea portului **PB** ca port de intrare, cu citirea periodică a poziției comutatoarelor și afișarea pe ecran a numărului respectiv:

```
200 LET DB=63: LET CB=127
```

```
210 OUT CB, 255: OUT CB, 255
```

```
300 LET X=IN DB
```

```
310 PRINT "X"; X
```

```
320 GO TO 300
```

Program pentru citirea periodică a stărilor comutatoarelor și afișarea rezultatului la diodele luminescente:

```
10 LET DA=31: LET CA=95
```

```
20 OUT CA, 225: OUT CA, 0
```

```
30 OUT DA, 0
```

```
40 LET DB=63: LET CB=127
```

```
50 OUT CB, 255: OUT CB, 255
```

```
400 LET X=IN DB
```

```
410 OUT DA, X
```

```
420 GO TO 400
```

În continuare se va prezenta cuplarea unei scheme de comandă a unui micromotor de curent continuu la biții **PA0** și **PA1**, ai portului de ieșire **PA**.

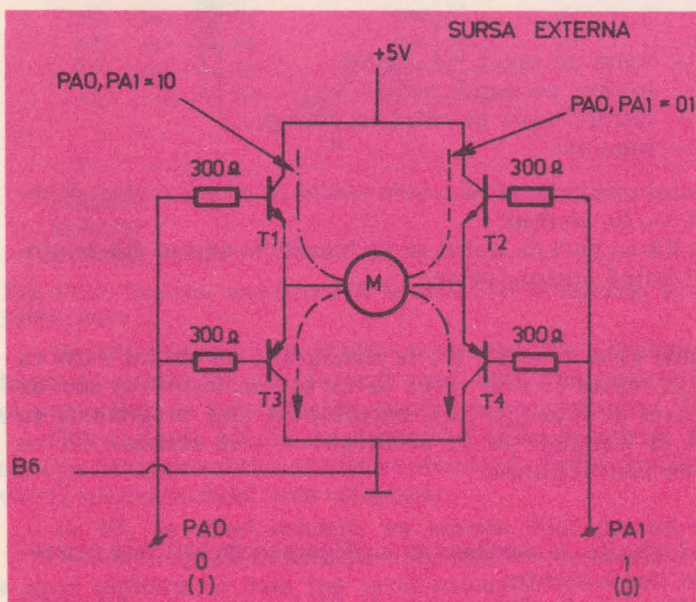


Fig. 13.11. Schema de comandă a unui micromotor de CC

Schema circuitului de comandă este dată în figura 13.11. Ea constă din puntea realizată cu tranzistoare *nnp* și *pnp* (BD 137 și BD 149), alimentate de la o sursă separată de +5V, care trebuie să asigure un curent de circa 300 mA. Bazele tranzistoarelor sunt comandate prin intermediul unor rezistențe de 300Ω, direct de la biții **PA0** și **PA1**, ai portului de ieșire **PA**.

Analiza schemei de comandă pune în evidență modul de operare:

- **PA0, PA1=00** — tranzistoarele **T1** și **T2** sunt blocate: motor nealimentat;
- **PA0, PA1=01** — tranzistoarele **T1** și **T4** sunt blocate, tranzistoarele **T2** și **T3** conduc: motor alimentat;



- PA0, PA1=10 — tranzistoarele T3 și T2 sînt blocate, tranzistoarele T1 și T4 conduc: motor alimentat;
- PA0, PA1=11 — tranzistoarele T3 și T4 sînt blocate: motor nealimentat.

În continuare se prezintă un program de comandă a sensului de rotație a motorului:

1 PRINT "Cele două programe se	10 LET DA=31: LET CA=95
selecționează cu numerele 1 și	20 OUT CA, 255: OUT CA, 0
2 după cum urmează"	30 OUT DA, 0
2 PRINT AT 5, 0 "Schimbarea sen-	35 RETURN
sului de rotație al motorului	40 LET DB=63: LET CB=127
din comutatoarele: 0 și 1"	50 OUT CB, 255: OUT CB, 255
3 PRINT AT 8, 0 "Schimbarea sen-	100 LET X=IN DB
sului de rotație al motorului	110 OUT DA, X
prin program"	120 GO TO 100
4 GO SUB 10: INPUT U: IF U=1	300 OUT DA, 1
THEN GO TO 40	310 PAUSE 100
5 IF U=2 THEN GO TO 300	320 OUT DA, 2
6 PRINT "Selectați corect pro-	330 PAUSE 200
gramele": GO TO 1	340 GO TO 300

### 13.3. Determinarea simultană a două constante fizice ( $\sigma$ și $g$ ) printr-un experiment unic, asistat de HC-85 \*)

În lucrarea de față se propune o îmbunătățire a metodei de determinare a coeficientului de tensiune superficială a unui lichid prin preluarea și prelucrarea în timp real a datelor experimentale de către microcalculator. De asemenea, din aceleași date se determina prin calcul și accelerația gravitațională locală.

*Principiul fizic al metodei de determinare a coeficientului de tensiune superficială a unui lichid.* Coeficientul de tensiune superficială ( $\sigma$ ) este o mărime de material, numeric egală cu rezultanta forțelor tangențiale care se exercită pe unitatea de lungime, la suprafața unei membrane de lichid.

Curgerea unui lichid printr-un tub vertical depinde esențial de tensiunea superficială a lichidului. Astfel, se știe că atunci cînd un lichid curge dintr-un

---

\* Lucrare prezentată de elevul Suciu Ioan Alexandru, cls. a-X-a, Liceul „Dimitrie Cantemir”, București, la Sesiunea de referate și comunicări a elevilor, Municipiul București, mai 1987.



tub vertical cu orificiu îngust are loc o curgere intermitentă, prin picături. Ruperea picăturii este un fenomen dinamic, complex. Dacă curgerea se face lent astfel încât să fie îndeplinite permanent condițiile de echilibru static, atunci se poate considera că picăturile care se formează succesiv, au mase aproximativ egale. Masa fiecărei picături depinde de tensiunea superficială a lichidului, de raza deschiderii tubului și de alți factori.

În teoria elementară a acestui fenomen se considera ca forță care se opune căderii picăturii și care echilibrează greutatea acesteia, în momentul ruperii, este tocmai tensiunea superficială a lichidului.

Dacă  $r$  este raza gîturii picăturii în porțiunea unde aceasta se rupe, iar  $m$  masa picăturii formate, atunci în momentul desprinderii acesteia, greutatea picăturii depășește tensiunea superficială. La limita se obține:

$$m \cdot g = 2 \cdot \pi r \cdot \sigma \quad (1)$$

Relația (1) permite determinarea lui  $\sigma$  dacă se cunosc masa unei picături și raza picăturii.

Experimental masa unei picături este mai greu de determinat, astfel încât determinarea lui  $\sigma$  se face după cum urmează.

Se considera două lichide: unul cu coeficientul de tensiune superficială cunoscut, celălalt — lichidul de cercetat.

Din același volum  $V$  se formează  $n_1$  picături din primul și  $n_2$ , din al doilea lichid. Aplicînd relația (1) în acest caz, pentru cele două lichide, se obține:

$$m_1 \cdot g = 2 \cdot \pi r \cdot \sigma_1 \quad (2)$$

respectiv,

$$m_2 \cdot g = 2 \cdot \pi \cdot r \cdot \sigma_2 \quad (3)$$

Masele  $m_1$  și  $m_2$  se determină din relațiile:

$$m_1 = \rho_1 \cdot V / n_1, \quad m_2 = \rho_2 \cdot V / n_2 \quad (4)$$

$\rho_1$  și  $\rho_2$  fiind densitățile celor două lichide.

Din (2), (3) și (4), în urma unor calcule elementare rezultă:

$$\sigma_1 = (\rho_1 \cdot n_2 / \rho_2 \cdot n_1) \cdot \sigma_2 \quad (5)$$

Relația (5) permite determinarea tensiunii superficiale a lichidului de cercetat, dacă se determină experimental numărul de picături care se formează din volume egale, din cele două lichide.

Deoarece  $n_2 \cdot \sigma_2 / \rho_2 = k$  este o constantă pentru un aparat dat și un lichid etalon dat, relația (5) devine:

$$\sigma_1 = k \cdot (\rho_1 / n_1) \quad (6)$$

Ca lichid etalon, de obicei, se ia apa distilată cu:

$$\rho_2 = 10^3 \text{ kg/m}^3 \quad \text{și} \quad \sigma_2 = 73,26 \cdot 10^{-3} \text{ N/m}$$



În cazul instalației realizate de autor, constanta are valoarea:  $439,56 \cdot 10^{-6}$ .

*Principiul fizic al metodei de determinare a accelerației gravitaționale locale.*  
Pe prima porțiune pictura de lichid cade liber în câmpul gravitațional al pământului. Din legea de mișcare rezultă expresia accelerației gravitaționale locale.

$$g = 2 \cdot h / \epsilon^2$$

unde  $h$  este spațiul parcurs de picătura în cădere. În experimentul realizat  $h = 20$  cm. Durata căderii s-a notat cu  $\epsilon$ .

Valoarea lui  $\epsilon$  rezultă din relația:

$$\epsilon = t - n \cdot t_f,$$

unde: —  $t$  reprezintă durata experimentului,  
—  $t_f$  este timpul de formare al unei picături,  
—  $n$  este numărul de picături.

Timpul de formare al unei picături în cazul apei a fost determinat anterior și are valoarea  $t_f = 6,32$  s.

Dispozitivul experimental constă dintr-o pipetă cu capilar, îndoit la partea inferioară.

Tubul este prevăzut în partea superioară cu un balonaș, pe care sînt trasate două repere, pentru a marca un volum bine determinat. Acest dispozitiv este cunoscut sub numele de stalagmometru.

Schema întregului experiment este prezentată în figura 13.12 și cuprinde: stalagmometru (1), senzorul (2), interfața (3) și calculatorul (4).

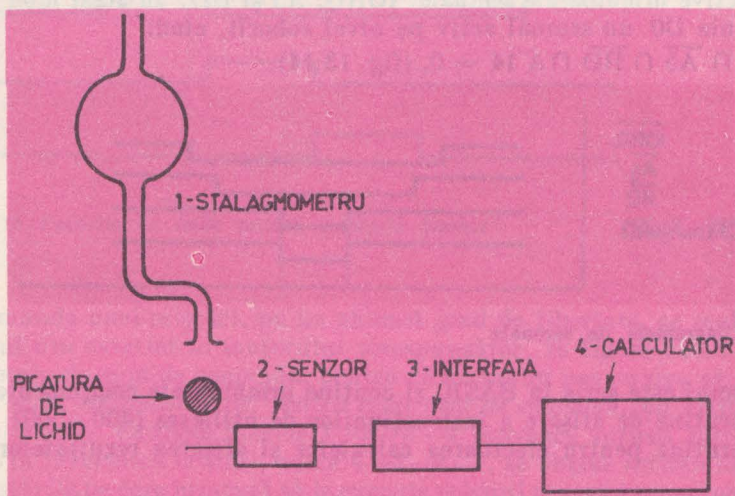


Fig. 13.12. Schema instalației

Partea de hardware este un dispozitiv care anunță calculatorul cînd a căzut o picătură. Instalația cuprinde un traductor care simulează închiderea tastei CR (ENTER), cînd a căzut o picătură, și o interfață. Simularea se realizează prin închiderea unui contact între linia de adresă A14 și linia de date D0 (fig. 13.13). Interfața are rolul de a introduce semnalul în calculator atunci

### 13. INTERFEȚE PENTRU MĂSURĂRI ȘI ROBOȚI



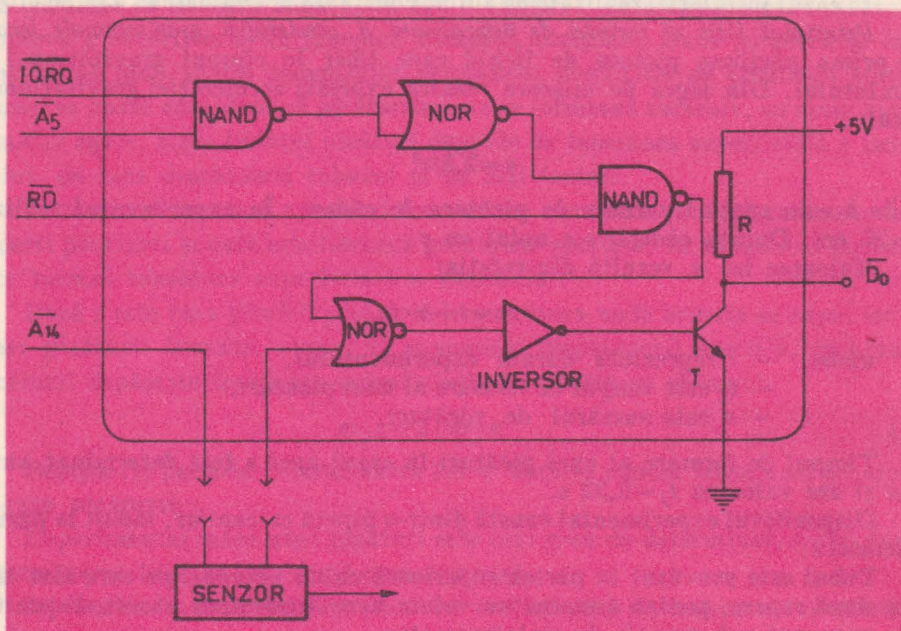


Fig. 13.13. Schema interfeței

cînd sînt active următoarele semnale:  $\overline{IORQ}$ ,  $\overline{A5}$  și  $\overline{RD}$ . În acest mod se forțează pe linia D0 un semnal activ pe nivel coborît, cînd:

$$\overline{IORQ} \cap \overline{A5} \cap \overline{RD} \cap A14 = 0, \text{ (fig. 13.14).}$$

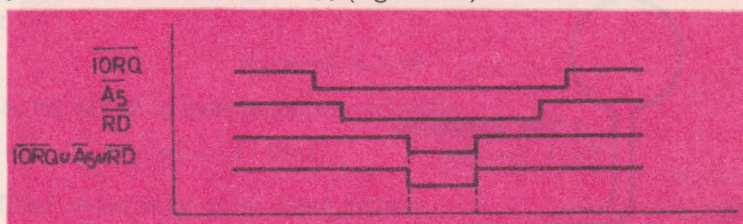


Fig. 13.14. Diagramele de semnale

Programul este scris în BASIC și conține următoarele componente:

- subrutina de afișare a instrucțiunilor de utilizare (400—500)
- subrutina pentru efectuarea calculelor și afișarea rezultatelor (300—320),

— rutina pentru contorizarea picăturilor și determinarea timpului total.

Rezultate. Instalația prezentată a fost testată pe mai multe lichide (apă, alcool, benzen) pentru care s-au obținut rezultate care, în cazul tensiunii superficiale, nu diferă cu mai mult de 4% de cele din literatură. Valorile obținute pentru accelerația gravitațională au fost în cadrul ordinului de mărime al acesteia. Rezultatele obținute confirmă superioritatea metodei propuse față de metoda clasică: viteza de lucru crește cu două ordine de mărime, iar eroarea care afectează măsurătoarea se reduce.



### 13.4. Conducerea minirobotului RIP 0.2 cu microcalculatorul HC-85\*.

Robotul RIP 0.2, construit de întreprinderea Automatica, este prevăzut cu șase motoare pas cu pas. Prin comanda acestor motoare se pot efectua mișcări suficient de complexe. Fiecare motor are patru faze, care sînt accesibile pentru comandă. Pentru fiecare motor există opt poziții alocate înfășurărilor celor patru faze (fig. 13.15).

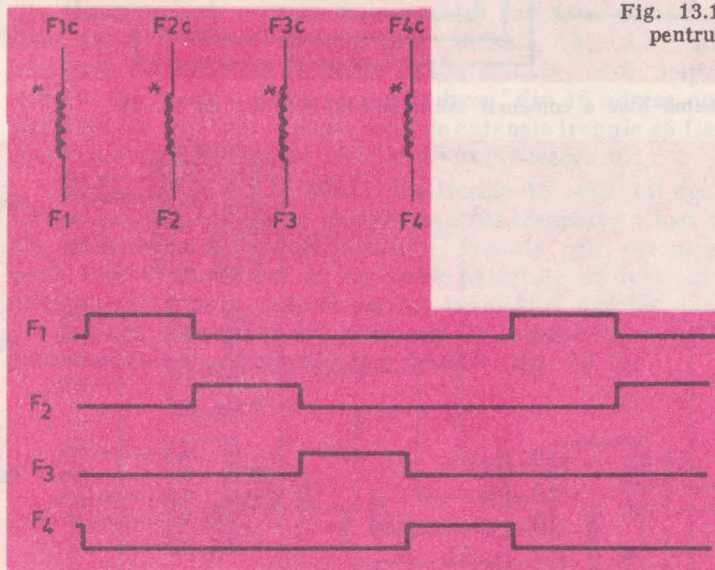


Fig. 13.15. Înfășurările fazelor pentru motorul pas cu pas

Fig. 13.16. Formele de unde pentru comanda fazelor

Comanda unei mișcări, pe un anumit grad de libertate, se realizează prin controlul din exterior al motorului corespunzător. În figura 13.16 se prezintă un exemplu de mod de comandă, pentru ca un motor pas cu pas să se rotească într-un sens. Se constată că, pentru acest mod de comandă, nu există două faze active simultan. Ordinea fazelor 1—2—3—4 provoacă rotirea într-un anumit sens. Prin inversarea ordinii, adică 4—3—2—1, se obține rotirea în sens opus.

Mișcările pe care urmează să le execute robotul necesită numeroase secvențe de impulsuri, care trebuie aplicate motoarelor robotului. Complexitatea secvențelor de comandă face necesară prezența unui element „inteligent” pentru conducerea robotului. În numeroase cazuri se realizează structuri bazate pe microprocesor, pentru conducerea robotului. Există sisteme de conducere a roboților care au cîte un microprocesor alocat fiecărui motor.

\* Autor asistent ing. Romulus Andrei, de la Catedra de Calculatoare, Facultatea Automatică, Institutul Politehnic București.



În cazul de față s-a utilizat microcalculatorul personal HC-85.

Între microcalculator și robot se plasează o interfață, cu ajutorul căreia se asigură semnalele electrice necesare comenzii fiecărui motor. Schema bloc a sistemului de conducere a robotului RIP 0.2, cu ajutorul microcalculatorului HC-85 este prezentată în figura 13.17

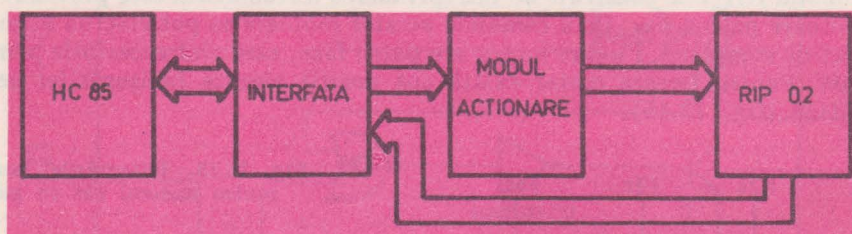


Fig. 13.17. Schema bloc a comenzii minirobotului RIP 0.2 cu HC-85

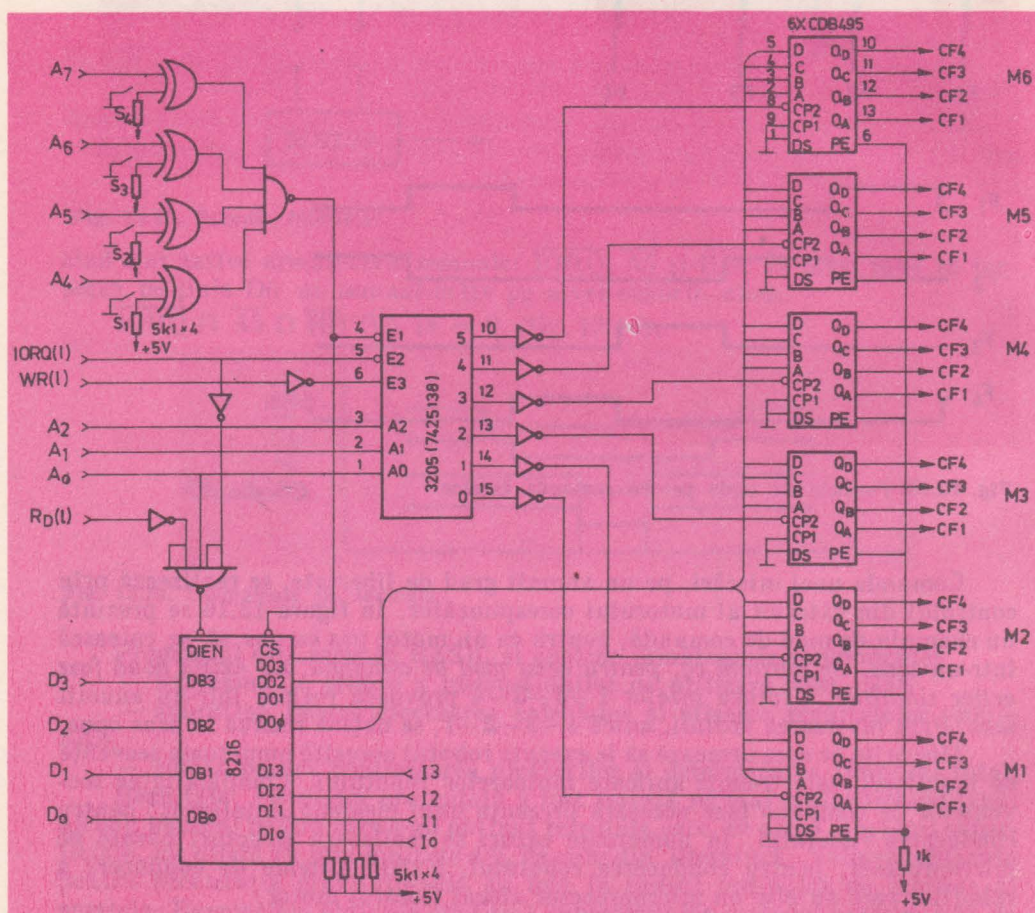


Fig. 13.18. Interfața cu minirobotul și schema electrică



**Interfața microcalculator-robot.** Interfața este adresată de microprocesor, în spațiul porturilor de intrare/ieșire, cu instrucțiuni **OUT**. Ea are rolul de a prelua de la calculator cuvintele de comandă pentru fazele fiecărui motor, de a le memora și a le transmite mai departe blocului amplificator. De asemenea, mai poate fi citită starea microîntreruptoarelor — indicatoare de limite de cursă — de pe robot. Schema interfeței este dată în figura 13.18. În continuare se va prezenta succint rolul celor 12 circuite integrate utilizate.

Registrele CDB 495 memorează starea fazelor, câte un registru pentru fiecare motor. Informația se poate înscrie în registre folosind instrucțiunea **OUT**. Întrucât acest tip de registre nu are prevăzută o intrare **RESET**, este necesar, ca la începutul operării, să se realizeze inițializarea lor prin program.

Interfața este controlată în spațiul adreselor de intrare/ieșire, cu instrucțiuni **IN/OUT**. Vor fi decodificate adresele **A0**, ..., **A7**. Adresele **A7**, ..., **A4** sînt introduse într-un bloc de porți SAU-Exclusiv, care, împreună cu comutatoarele **S4**—**S1**, permit alegerea unei adrese, din 16 adrese posibile. Circuitele folosite pentru legătura la conectorul de extensie trebuie să fie de tip LS, deoarece sînt comandate direct de ieșirile microprocesorului.

Semnalele **IORQ(1)**, **WR(1)** și adresele **A2**—**A0** sînt decodificate de circuitul 3205 (sau 74LS138), cînd decodificatorul respectiv a fost selectat prin adresele **A7**, ..., **A4**. Ieșirile decodificatorului, trecute prin circuite inversoare, controlează înscrierea datelor în registrele selectate de instrucțiune. Circuitul bidirecțional 8216 este utilizat pentru transferul datelor de la/calculator.

Ieșirile registrelor 495 comandă 24 scheme identice (6 motoare  $\times$  4 faze) pentru acționarea înfășurărilor fazelor (fig. 13.19).

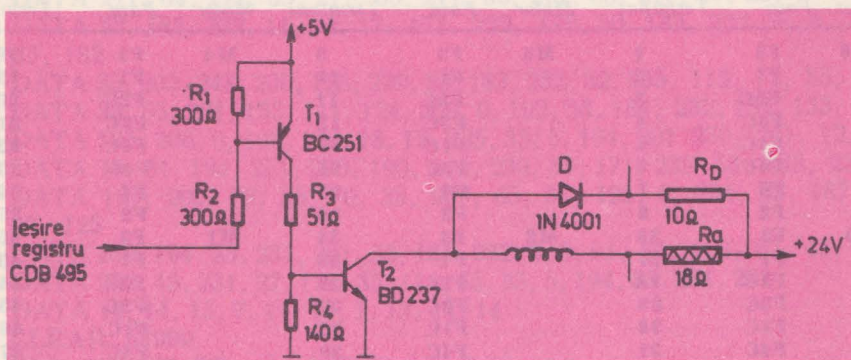


Fig. 13.19. Schema de acționare pentru faza unui motor

Pentru fiecare grup de 4 faze (un motor) s-a utilizat o rezistență cu rol de accelerare.

**Modul de funcționare:** cînd ieșirea registrului 495, de exemplu **CF4**, trece în zero, se va satura tranzistorul **BC251**, care va conduce la intrarea în saturație a tranzistorului **BD237**. Ultimul tranzistor va comanda, în acest mod, circulația curentului prin înfășurarea fazei respective. De menționat că pentru comanda unei faze, calculatorul trebuie să trimită valoarea zero pe bitul corespunzător, din registrul 495. Astfel, pentru rotirea într-un anumit sens a unui motor, trebuie trimisă secvența:



7 (0111)  
11 (1011)  
13 (1101)  
14 (1110)

care va asigura activarea pe rînd a fiecărei faze. Timpul de menținere a curentului prin înfășurarea unei faze este dependent de tipul motorului utilizat.

*Robotul RIP 0.2.* Acest microrobot face parte din categoria microroboților din generația a-II-a și poate fi folosit pentru verificarea algoritmilor și programelor roboților industriali.

RIP 0.2 este constituit dintr-o bază imobilă și patru segmente mobile: turela, umăr, cot, încheietura mîinii.

Încheietura mîinii s-a realizat cu un mecanism diferențial, care permite înclinarea și rotirea „gripper-ului”

Construcția robotului permite combinarea mișcărilor corpului, umărului, cotului și încheieturii mîinii. Mișcarea fiecărei articulații este controlată de un cablu flexibil, care pornește de la blocul motor aflat în turelă și ajunge la încheietura arborelui respectiv.

Pentru fiecare grad de libertate, blocul de acționare este alcătuit dintr-un motor, un reductor și o roată de cablu. De la fiecare roată pleacă un cablu tensionat, care trece peste un scripete, ajungînd la elementul care trebuie mișcat, întorcîndu-se apoi la roata de cablu.

Cele patru faze de la fiecare motor sînt disponibile la un conector cu 50 de contacte conform cu lista de mai jos:

<i>Motor</i>	<i>Faza</i>	<i>Terminal</i>	<i>Motor</i>	<i>Faza</i>	<i>Terminal</i>	<i>Motor</i>	<i>Faza</i>	<i>Terminal</i>
M6	F3	1	M5	F3	9	M4	F3	18
	F1	2		F1	10		F1	19
	F2C	3		F2C	11		F2C	20
	F3C	4		F3C	12		F3C	21
	F1C	5		F1C	13		F1C	22
	F4C	6		F4C	14		F4C	23
	F4	7		F4	15		F4	24
	F2	8		F2	16		F2	25
M3	F3	26	M2	F3	34	M1	F3	42
	F1	27		F1	35		F1	43
	F2C	28		F2C	36		F2C	44
	F3C	29		F3C	37		F3C	45
	F1C	30		F1C	38		F1C	46
	F4C	31		F4C	39		F4C	47
	F4	32		F4	40		F4	48
	F2	33		F2	41		F5	49

*Programarea mișcării robotului.* Pentru început s-a realizat un nucleu de program în BASIC, împreună cu rutinele în limbaj de asamblare, care permit unui operator nespecializat să poată lansa comenzi pentru robot.

La lansarea în execuție a programului, pe ecranul televizorului este afișat mesajul:

#### OPȚIUNI

*t* = turelă  
*u* = umăr  
*b* = braț  
*m* = mînal

*n* = mîna2  
*s* = stringe/desface  
*g* = sfîrșit segment  
*p* = sfîrșit program

## VI. HC-85 IN PROCESE. PACHETE PROGRAME



## SEGMENTUL NUMĂRUL 1

În cadrul unui segment se specifică elementele care urmează a fi mișcate. Pentru exemplificare, să presupunem că s-a apăsât tasta „u”. În acest caz ecranul este șters și se afișează în continuare mesajul:

Umăr

sus=s,      jos=j,      a=anulare      c=da.

Se apasă tasta „a” dacă nu se dorește mișcarea umărului sau una din literele „s”, „j” în mod corespunzător. Să presupunem că s-a apăsât „s”. Atunci, în continuare, se cere introducerea numărului de pași, cu care va fi mișcat umărul.

Număr pași (1,...255)=

după care se revine în meniul cu opțiuni.

În continuare se poate programa alt element, în cadrul segmentului 1, sau se poate trece la alt segment, ori la execuție.

În cadrul unui segment elementele se mișcă simultan. Segmentul următor începe numai după terminarea tuturor mișcărilor din segmentul curent.

La terminarea tuturor segmentelor, se poate relua întregul ciclu.

```
1 OUT 64, 15: OUT 65, 15: OUT 66, 15: OUT 67, 15: OUT 68, 15: OUT
69, 15
10 DATA 17, 185, 232, 42, 169, 232, 14, 18, 126, 18, 35
12 DATA 19, 13, 194, 128, 230, 175, 50, 184, 232, 205, 187, 230, 58, 183, 232
14 DATA 167, 202, 172, 230, 205, 2, 231, 205, 20, 231, 33, 184, 232, 52
16 DATA 126, 254, 4, 194, 140, 230, 205, 30, 231, 195, 136, 230, 62, 15
18 DATA 211, 64, 211, 65, 211, 66, 211, 66, 211, 68, 211, 69, 201
20 DATA 33, 185, 232, 14, 64, 17, 171, 232, 175, 50, 183, 232, 6, 6, 126,
35, 182
22 DATA 35, 202, 248, 230, 126, 229, 33, 183, 232, 52, 235, 113, 12, 35, 235
24 DATA 33, 55, 231, 133, 111, 124, 206, 0, 103, 58, 184, 232, 135, 133, 111
26 DATA 124, 206, 0, 103, 126, 18, 19, 225, 35, 5, 194, 201 230, 201, 12, 62
28 DATA 10, 61, 194, 251, 230, 195, 242, 230, 33, 171, 232, 58, 183, 232
30 DATA 167, 200, 78, 35, 70, 35, 237, 65, 61, 195, 9, 231, 33, 187, 0,
43, 125
32 DATA 180, 194, 23, 231, 201, 33, 185, 232, 6, 6, 94, 35, 126, 87, 43, 179
34 DATA 202, 45, 231, 27, 115, 35, 114, 35, 35, 5, 194, 35, 231, 201
36 DATA 14, 14, 13, 7, 11, 11, 7, 13, 14, 14
60 CLEAR 58999
62 CLS: PRINT "Execut initializări..."
65 FOR i=0 TO 200
70 READ k: POKE 59000+i, k
80 NEXT i
98 LET tabert=59561
99 LET tabinfo=59201
100 CLS : LET nrseg=1
102 FOR i=0 TO 359
103 PRINT " ";
104 POKE tabinfo+i,0
105 NEXT i
110 GO SUB 1500
115 PRINT AT 14, 4; "Segment 1 numărul" ;nrseg
```



```

120 IF INKEY$="b" THEN GO TO 240
130 IF INKEY$="u" THEN GO TO 380
140 IF INKEY$="t" THEN GO TO 510
150 IF INKEY$="m" THEN GO TO 900
160 IF INKEY$="n" THEN GO TO 770
170 IF INKEY$="s" THEN FO TO 640
180 IF INKEY$="g" THEN GO TO 1030
190 IF INKEY$="p" THEN GO TO 1160
200 GO TO 120
240 CLS
250 PRINT AT 4, 4; "Brat"
260 PRINT AT 7, 2; "sus=s, jos=j"
265 PRINT AT 9, 2; "a=anulare comandă"
270 IF INKEY$="s" THEN GO TO 310
280 IF INKEY$="j" THEN GO TO 320
290 IF INKEY$="a" THEN GO TO 110
300 GO TO 270
310 POKE tabinfo+18*(nrseg-1)+2,0: GO TO 330
320 POKE tabinfo+18*(nrseg-1)+2,1
330 GO SUB 1700
350 POKE tabinfo+18*(nrseg-1), b1
360 POKE tabinfo+18*(nrseg-1)+1, b2
370 GO TO 110
380 REM umăr
390 CLS
400 PRINT AT 4, 4; "Umăr"
410 PRINT AT 7, 2; "sus=s, jos=j"
415 PRINT AT 9,2; "a=anulare comandă"
420 IF INKEY$="s" THEN GO TO 460
430 IF INKEY$="j" THEN GO TO 470
440 IF INKEY$="a" THEN GO TO 110
450 GO TO 420
460 POKE tabinfo+18*(nrseg-1)+5, 1: GO TO 480
470 POKE tabinfo+18*(nrseg-1)+5,0
480 GO SUB 1700
490 POKE tabinfo+18*(nrseg-1)+3, b1
495 POKE tabinfo+18*(nrseg-1)+4, b2
500 GO TO 110
510 REM Turela
520 CLS
530 PRINT AT 4, 4; "Turela"
540 PRINT AT 7,2; "stinga=s, dreapta=d"
545 PRINT AT 9, 2; "a=anulare comandă"
550 IF INKEY$="s" THEN GO TO 590
560 IF INKEY$="d" THEN GO TO 600
570 IF INKEY$="a" THEN GO TO 110
580 GO TO 550
590 POKE tabinfo+18*(nrseg-1)+8, 0: GO TO 610
600 POKE tabinfo+18*(nrseg-1)+8, 1
610 GO SUB 1700

```



```

620 POKE tabinfo+18*(nrseg-1)+6, b1
625 POKE tabinfo+18*(nrseg-1)+7, b2
630 GO TO 110
640 REM stringe desface
650 CLS
660 PRINT AT 4, 4; "Închide/Deshide"
670 PRINT AT 7,2; "inchide=i, deschide=d"
675 PRINT AT 9,2; "a=anulare comandă"
680 IF INKEY$="d" THEN GO TO 720
690 IF INKEY$="i" THEN GO TO 730
700 IF INKEY$="a" THEN GO TO 110
710 GO TO 680
720 POKE tabinfo+18*(nrseg-1)+11,0: GO TO 740
730 POKE tabinfo+18*(nrseg-1)+11,1
740 GO SUB 1700
750 POKE tabinfo+18*(nrseg-1)+9, b1
755 POKE tabinfo+18*(nrseg-1)+10, b2
760 GO TO 110
770 REM Mina 2
780 CLS
790 PRINT AT 4,4; "Mina 2"
800 PRINT AT 7,2; "sus=s, jos=j"
805 PRINT AT 9,2; "a=anulare comandă"
810 IF INKEY$="s" THEN GO TO 850
820 IF INKEY$="j" THEN GO TO 860
830 IF INKEY$="a" THEN GO TO 110
840 GO TO 810
850 POKE tabinfo+18*(nrseg-1)+14,0: GO TO 870
860 POKE tabinfo+18*(nrseg-1)+14,1
870 GO SUB 1700
880 POKE tabinfo+18*(nrseg-1)+12, b1
890 POKE tabinfo+18*(nrseg-1)+13, b2
895 GO TO 110
900 REM Mina 1
910 CLS
920 PRINT AT 4,4; "Mina 1"
930 PRINT AT 7,2; "sus=s, jos=j"
940 PRINT AT 9,2; "a=anulare comandă"
945 IF INKEY$="s" THEN GO TO 980
950 IF INKEY$="j" THEN GO TO 990
960 IF INKEY$="a" THEN GO TO 110
970 GO TO 945
980 POKE tabinfo+18*(nrseg-1)+17,1: GO TO 1000
990 POKE tabinfo+18*(nrseg-1)+17,0
1000 GO SUB 1700
1010 POKE tabinfo+18*(nrseg-1)+15, b1
1015 POKE tabinfo+18*(nrseg-1)+16, b2
1020 GO TO 110
1030 REM Sfârșit pas
1040 IF nrseg=20 THEN GO TO 1090

```

### 13. INTERFEȚE PENTRU MĂSURĂRI ȘI ROBOȚI



```

1050 CLS
1060 PRINT AT 4,4; "Urmează pasul numărul"; nrseg+1
1070 LET nrseg=nrseg+1
1080 PAUSE 50: GO TO 110
1090 CLS
1095 PRINT AT 4,4; "S-a introdus numărul"
1100 PRINT AT 5,4; "maxim de segmente"
1105 PRINT AT 7,3; "a=anulează ultimul pas"
1110 PRINT AT 9,3; "p=sfîrșit introducere date"
1115 PRINT AT 10,3; "și începe execuție program"
1130 IF INKEY$="a" THEN GO TO 110
1140 IF INKEY$="p" THEN GO TO 1160
1150 GO TO 1130
1160 CLS
1166 PRINT AT 4,4; "Numărul reluări=";: INPUT nr
1167 PRINT AT 2,4; "Reluarea numărul ";i
1170 PRINT AT 10,4; "Nr. segmente="; nrseg
1177 FOR k=1 TO nrseg
1180 PRINT AT 4,4; "Se executa segmentul"
1185 PRINT AT 5,4; "numărul ";k;" "
1188 LET n=tabinfo+18*(k-1)
1190 POKE tabcrt+1, INT (n/256)
1195 POKE tabcrt, n-256 INT (n/256)
1200 PRINT USR 59000
1205 PRINT AT 6,0;" "
1210 NEXT k
1215 NEXT i
1220 PRINT AT 12,4; "Execuție terminată"
1230 PRINT AT 18,4; "Reiau execuția? (d=da, n=nu)"
1240 IF INKEY$="d" THEN GO TO 1167
1250 IF INKEY$="n" THEN GO TO 100
1260 GO TO 1240
1500 REM șterge ecran+mesaje
1510 CLS
1520 PRINT AT 2,6; "OPȚIUNI"
1530 PRINT AT 4,4; "t=turela"
1540 PRINT AT 5,4; "u=umăr"
1550 PRINT AT 6,4 "b=braț"
1560 PRINT AT 7,4; "m=mîna1"
1570 PRINT AT 8,4; "n=mîna2"
1580 PRINT AT 9,4; "s=îchide/deschide"
1584 PRINT AT 10,4; "g=sfîrșit segment"
1586 PRINT AT 11,4; "p=sfîrșit program"
1590 RETURN
1700 PRINT AT 12,2; "Număr pași (0..65535)=";
1710 INPUT n
1720 IF n>65535 OR n<0 THEN GO TO 1700
1730 LET b2=INT(n/256)
1740 LET b1=(INT n)-256*62
1750 RETURN

```

## VI. HC-85 IN PROCESE, PACHETE PROGRAME



Există o serie de programe, generalizabile realizate în țară sau în strălănatate, pentru calculatoarele de tip HC-85 (compatibile SINCLAIR-SPECTRUM). În continuare ne vom referi la cîteva tipuri strălăne, adoptate în țară. Ele sînt stocate pe medii magnetice și cunosc o lărgă răspîndire; în anexe sînt listate și alte programe.

## 14.1. Program pentru grafică în trei dimensiuni

Programul pentru grafică în trei dimensiuni, numit VU-3D,\* are un caracter relativ complex, permițînd ca, pe baza unui set de comenzi, prezentat sub forma unui meniu pe ecran, să se poată genera imaginile unui obiect sau grup de obiecte, imagini care se stochează în memoria calculatorului și opțional — pe caseta magnetică, sub forma de fișiere de date.

Rezultatele obținute pot fi afișate pe ecran sau la o imprimantă grafică, capabilă să reproducă imaginea de pe ecran.

Asupra obiectelor astfel generate se pot efectua diferite operații de mărire, reducere, deplasare: stînga-dreapta, sus-jos, afișarea tuturor liniilor care concură la realizarea desenului, afișarea liniilor ascunse, afișarea zonelor umbrite.

Utilizatorul poate observa obiectul sau grupul de obiecte din diferite unghiuri, de la diferite distanțe, se poate deplasa în jurul lor etc.

După încărcarea programului VU-3D, de pe casetă, apar opțiunile de creare, (CREATE —1) a unui nou obiect sau de încărcare (LOAD-2) a unui fișier de date de pe casetă.

În cazul selectării opțiunii CREATE, utilizatorul poate desena, cu ajutorul unui set de comenzi, un nou obiect în trei dimensiuni.

În starea CREATE programul afișează, în partea superioară a ecranului, un meniu format din trei linii ce conțin comenzi, care pot fi apelate prin tasta-re inițialelor. La baza ecranului se află linia de date.

Operația de creare se bazează pe introducerea interactivă a unei secvențe de secțiuni prin obiectul care se proiectează. Secțiunile corespund unor planuri paralele cu XOY, avînd diferite valori pentru Z. Pe ecran sînt proiectate secțiunile în planul XOY, axa OX fiind la baza ecranului (de la stînga — la dreapta), iar axa OY, la marginea stînga a ecranului (de jos — în sus).

Este indicat ca, înaintea începerii desenării, utilizatorul să aibe deja o imagine asupra obiectului sau setului de obiecte pe care dorește să le creeze. În alegerea axei Z se recomandă ca aceasta să fie axa cu cea mai pronunțată simetrie a obiectului. Astfel, axa pentru construirea unei mingi de rugby trece prin punctele extreme ale acesteia. În același mod, axa Z a unui pahar cu picior va coincide cu axa centrală a piciorului.

\* VU-3D. PSION Comp. — 1982



O (Deschidere-Open). După selectarea opțiunii CREATE, utilizatorul va putea începe desenarea unei noi figuri, prin activarea comenzii Open (O). Aceasta va permite definirea primei secțiuni a obiectului. Activarea comenzii O va fi urmată de modificarea indicatorului comenzii și de apariția unui cursor pe ecran. Poziția cursorului este indicată în linia de date, de la baza ecranului. Prin activarea tastelor cu săgeți, cursorul se poate deplasa în orice poziție, pe ecran.

Prima secțiune a unui obiect ( $Z=0$ ) se definește printr-un set de linii care se închid (un poligon). Se poziționează cursorul în punctul de unde se dorește trasarea unui segment de dreaptă (o linie) și se activează tasta S pentru start. Se deplasează cursorul în punctul corespunzător celui alt capăt al segmentului și se activează tasta L, pentru desenarea liniei. Din nou se deplasează cursorul, la o altă poziție și se activează tasta L. Se continuă secvența de operații pînă cînd s-a terminat desenarea figurii dorite. Pentru a închide figura, în planul dat, se activează tasta E (End) și utilizatorul revine în funcția CREATE. Dacă în starea Open s-a comis o eroare, ultima linie desenată se poate șterge prin activarea tastei D (Delete).

Într-o secțiune pot fi incluse mai multe obiecte. Pentru a începe desenarea altui obiect sau a unei zone goale în obiectul desenat, în aceeași secțiune, se activează tasta O (Open).

Liniile unei figuri nu trebuie să se intersecteze. De asemenea, două figuri nu se pot intersecta. Starea Open va detecta tentativele de intersectare și nu va permite desenarea unei noi linii, pînă cînd cursorul nu a fost deplasat sau pînă cînd ultima/ultimele linii nu au fost modificate în mod corespunzător. Același lucru se va petrece și în cazul în care linia terminală a unei figuri, la invocarea comenzii de terminare End, va intersecta o linie existentă.

În starea CREATE, fie prin comanda Open, fie prin repetarea figurii din planul anterior ( $Z$  anterior), se pot include în noul plan mai multe figuri simultan. Cu ajutorul comenzilor M (Mărire — Magnify), R (Reducere — Reduce) și a deplasărilor cu tastele marcate cu săgeți, fiecare figură poate fi mărită, redusă sau deplasată separat. Alegerea unei figuri din serie se realizează prin tastarea comenzii F (Figură — Figure). Activarea repetată a tastei F permite selecția figurii dorite, prin afișarea ei cu linii întrerupte, în opoziție cu figurile neselectate, care sînt prezentate cu linii continue.

M (Mărire — Magnify). O figură care a fost desenată după comanda Open sau repetată, prin trecerea la următorul plan — alta valoare pentru  $Z$ , poate fi mărită activînd tasta M. Menținerea tastei M activate, asigură mărirea continuă a figurii, pînă la dimensiunile dorite.

R (Reducere — Reduce). Comanda Reducere are ca efect micșorarea dimensiunilor figurii, cît timp este activată tasta R.

Deplasarea figurii în plan, la stînga, în jos, în sus, la dreapta se realizează acționînd simultan tastele CS și 5/6/7/8.

N (Următoarea valoare pentru  $Z$  — Next  $Z$ ). După terminarea figurilor dintr-un plan dat (o anumită valoare pentru  $Z$ ), activînd tasta N se obțin noi valori pentru  $Z$ , care sînt afișate la baza ecranului. Figurile din planul anterior de lucru se repetă automat și în noul plan. Utilizatorul le poate modifica în mod corespunzător, prin comenzile Reducere/Mărire și/sau deplasare.

O minge de rugby poate fi desenată începînd cu un poligon cu 12 laturi ( $Z=0$ ), care va fi redus pentru a marca una din extremitățile obiectului. În continuare, trecînd la alte valori ale lui  $Z$ , se vor efectua măriri treptate, pînă la



secțiunea cu aria maximă, după care se vor face reduceri în mod gradat, până la atingerea celeilalte extremități.

C (Închidere — Close). Desenul unui obiect poate fi terminat, pentru orice valoare curentă a lui Z, prin activarea comenzii Close.

Pe parcursul desenării unui obiect se pot deschide noi desene, fie pentru alte obiecte, fie pentru goluri în obiectul dat, cu condiția ca să nu apară intersecții între secțiunile obiectelor.

F (Figura — Figure). În cazul existenței mai multor obiecte, terminarea unui obiect se realizează prin activarea tastei F, până cînd apare figura dorită, desenată cu linii întrerupte. Activînd comanda Close, desenul obiectului respectiv este terminat. Operația de desenare se continuă pentru obiectele care nu au fost selectate prin comanda F.

Q (Terminare — Quit). Pentru a termina desenarea tuturor obiectelor și a reveni în meniul principal, se activează comanda Quit.

Meniul principal dispune de următoarele opțiuni:

1. Modifică figura,
2. Abandonează,
3. Încarcă un fișier de date,
4. Creează o nouă figură,
5. Salvează un fișier de date,
6. Afișează.
7. Modifică culorile.

Opțiunile 1, 2, 3, 4, 5, și 7 nu necesită explicații suplimentare. Un interes deosebit îl prezintă opțiunea 6 (Afișare — Display).

După crearea unui obiect sau grup de obiecte cu ajutorul funcției CREATE, sau după încărcarea unui fișier de date (reprezentînd desenul unui obiect) se poate trece la afișarea pe ecran a obiectului în trei dimensiuni, pe baza a trei opțiuni: 1) reprezentarea tuturor liniilor, 2) diagrama liniilor ascunse, 3) diagrama hașurată a solidului.

Toate reprezentările sînt tridimensionale, în perspectivă.

Utilizatorul poate observa obiectul din orice poziție în spațiu, din interiorul/exteriorul obiectului, din orice unghi și de la orice distanță. Se poate considera că observatorul se deplasează pe suprafața unei sfere, avînd obiectele plasate foarte aproape de centru sau chiar în centru. Modificarea razei sferei se poate realiza prin comenzile: F (Mărire a razei sferei — Far) și N (Micșorare a razei sferei — Near).

Activînd simultan tastele CS și F/N, operațiile asociate se vor efectua cu viteză mică, pentru a asigura un control mai precis.

Deplasarea pe sferă a observatorului, respectiv-rotirea obiectului în diverse sensuri se realizează prin activarea tastelor CS și 5/6/7. și 8.

## 14.2. Baze de date

### 1. Introducere.

Programul Bază de date (Data base\*) reprezintă implementarea pe calculator a sistemului de manipulare a fișelor indexate, organizate pe cartele. Aceasta

\* DATA BASE Gemeni Marketing Limited 1982.



permite obținerea unor înregistrări privind diferite subiecte, pornind de la definirea informației, care se dorește a fi stocată în fișierul dat.

În continuare se va explica terminologia folosită în asociație cu baza de date, în comparație cu un sistem manipulat manual de cartele indexate.

**FIȘIER.** Un ansamblu complet de cartele constituie un fișier de date. Informația din acest fișier este stocată pe casetă și poate fi încărcată în calculator în orice moment, în vederea examinării sau modificării.

În cazul de față, atît programul de gestiune, cît și fișierul de date sînt stocate pe casetă ca un tot. Astfel, la încărcarea programului, toată informația care a fost în fișier, în momentul în care programul a fost stocat, va fi introdusă în memoria calculatorului.

**ÎNREGISTRARE.** O cartelă particulară în ansamblul de cartele reprezintă o înregistrare. Înregistrarea conține informații în funcție de modul în care au fost specificate de utilizator cîmpurile, care intră în componenta ei.

**CÎMP.** Cîmpurile reprezintă titlurile pentru fiecare cartelă. *În total pot exista 19 asemenea cîmpuri.* Ca exemplu se poate da înregistrarea unui angajat:

Nr.	Titlul	Tip	Lungimea
1	Nume	s	19
2	Vîrsta	n	2
3	Secția	s	9
4	Retribuția	n	5

*Tipul se refera la conținutul cîmpului, care poate fi text (s) sau număr (n).* Distincția este necesară deoarece calculele se pot efectua numai pe cîmpurile numerice, iar sortarea folosește rutine diferite pentru cîmpurile numerice sau cele ce conțin texte.

## 2. Încărcarea

Încărcarea programului se realizează cu ajutorul comenzii `LOAD"" CR`. În mod normal încărcarea de pe caseta durează 30".

## 3. Lista de opțiuni — (MENIU).

În vederea selecției unei opțiuni din listă, se tastează numărul opțiunii și apoi CR.

Pentru a reveni în lista de opțiuni, la ultimul cîmp introdus sau pentru a corecta o eroare, se tastează ".

## 4. Memoria.

În cazul unui calculator cu o memorie RAM de 16 Ko, fișierul de date poate ocupa cel mult 1 Ko, în timp ce, în cazul unei memorii RAM de 48 Ko, fișierul de date poate ocupa pînă la 20 Ko.

## 5. Adăugarea unei înregistrări la un fișier.

Dacă există înregistrări în fișier, se impune activarea rutinei de definire a cîmpurilor dorite. Este necesar ca, în prealabil, să se facă o structurare a cîmpurilor, pentru a stoca informația într-o manieră cît mai eficientă.

Se reamintește că tipul cîmpului poate fi s sau n și că lungimea sa maximă poate fi de 19 caractere alfa-numerice.



După crearea fișierului, opțiunea 1 va asigura afișarea secvențială a titlurilor fiecărui câmp. În continuare se vor completa elementele solicitate în fiecare câmp, după care se va tasta CR, pentru introducerea informației respective.

Activînd comanda *Escape* ( $\backslash$ ), se revine la câmpul anterior sau la meniul principal, dacă cursorul se află în primul câmp.

## 2 6. Modificarea unei înregistrări.

Această opțiune este practic asemănătoare cu prima opțiune, cu excepția că fiecare câmp este afișat împreună cu conținutul său. Activînd tasta CR (Enter), se va menține conținutul prezent al înregistrării; introducerea altei informații în prealabil va duce la modificarea câmpului respectiv.

## 3 7. Înlăturarea unei înregistrări.

Atunci cînd o înregistrare este înlăturată, toate celelalte înregistrări aflate deasupra ei vor fi deplasate în jos, modificîndu-se în mod corespunzător și numerele lor.

O înregistrare înlăturată este pierdută definitiv.

## 4 8. Regăsirea/Sortarea înregistrărilor.

După selectarea acestei opțiuni, se activează tasta *f*, pentru regăsirea unei înregistrări, sau tasta *s*, pentru sortarea înregistrărilor.

**REGASIREA.** Câmpurile sînt afișate pe ecran în vederea selecției câmpului dorit, prin introducerea valorii pe care trebuie să o găsească programul. Dacă de exemplu, se selectează câmpul 1, cu cheia de regăsire PETRE, programul va examina toate înregistrările care au în câmpul 1 primele cinci caractere identice cu cheia PETRE. Înregistrările sînt afișate una cîte una, prin activarea oricărei taste.

**SORTAREA.** Ca și pentru opțiunea de regăsire, se selectează câmpul dorit, întregul fișier fiind sortat în ordinea ascendentă a conținutului aceluia câmp.

În funcție de tipul câmpului, sortarea are loc automat și în mod diferit. Toate înregistrările sînt deplasate fizic, schimbîndu-se în mod corespunzător și numerele lor. Dacă în fișier sînt multe înregistrări, această operație necesită un anumit timp.

## 5 9. Listarea/Tipărirea înregistrărilor.

În acest scop se introduc numerele primei și ultimei înregistrări din zona fișierului care se dorește a fi afișată pe ecran (opțiunea *s*) sau tipărită (opțiunea *p*).

## 6 10. Numărul total de obiecte dintr-un câmp.

Această opțiune solicită introducerea numărului câmpului în cauză (câmpul trebuie să fie numeric), după care se va afișa totalul general al câmpurilor cu același titlu din întregul fișier.

## 7 11. Crearea unui nou fișier.

Această opțiune va lansa programul de la început, anulînd toată informația introdusă pînă la acel moment.

## 14. GRAFICĂ 3-D. BAZE DE DATE ș.a.



## 8 12. Stocarea fișierului pe casetă.

Opțiunea asigură stocarea pe casetă, atât a programului de gestiune a bazei de date, cât și a fișierului de date.

NOTA.

Dacă, din întâmplare, se comite o greșeală, fie prin selectarea opțiunii de ieșire din program, fie prin acționarea tastei BREAK, introducând comanda GO TO 1000, urmată de CR, se vor restarta programul și fișierul curent de date.

## 14.3. Tabelare electronică

### 1. Generalități.

VU-CALC\* este un program destinat analizei financiare, bugetului, calculului tabelelor științifice-ingineresti, analizei statistice etc. În esență el calculează și afișează tabele de numere și nume.

Operarea se bazează pe o tabelă sau grilă inițial vidă, compusă din compartimente organizate pe linii și coloane. Cu ajutorul unor comenzi simple se pot invoca expresii algebrice, formule etc., care leagă un compartiment de altul, o linie de alta sau o coloană cu alta, astfel încît, calculatorul poate actualiza rapid întreaga tabelă.

În compartimente particulare se pot introduce date sau nume, prin simpla schimbare a unui sau mai multor parametri, ceea ce asigură reevaluarea și afișarea tabelii, într-un timp foarte scurt, pentru diferite situații.

### 2. Prezentare (TABELA, CURSOR, FEREASTRA).

După încărcarea programului, în partea superioară a ecranului sînt afișate două linii de comandă, în mijlocul ecranului se prezintă o zonă albă (vidă), iar la baza ecranului este plasată linia de intrare. Zona albă, din mijlocul ecranului, reprezintă o fereastră din tabelă. O tabelă poate fi vizualizată ca o grupare de compartimente, organizate pe linii și coloane.

Liniile sînt marcate alfabetic, iar liniile curente, prezentate în fereastră sînt văzute din extrema stîngă.

Coloanele sînt marcate numeric începînd cu 01, iar coloanele curente prezente în fereastră au marcajul plasat pe linia imediat superioară ferestrei.

Fiecare compartiment este definit în mod unic prin litera asociată liniei și prin numărul coloanei. Astfel, A01 sau A1 se referă la compartimentul din colțul din stînga sus al tabelii.

În oricare din etapele de folosire a programului se utilizează un cursor, reprezentat printr-un dreptunghi mare, de culoare roșie, plasat în tabelă. În vederea citirii sau introducerii de date, text sau formule, cursorul poate fi deplasat în cele patru direcții cu ajutorul tastelor marcate cu săgeți (5, 6, 7, 8), acționate simultan cu tasta CS.

---

\* VU-CALC — PSION-Comp. 1982



Cînd cursorul ajunge la un compartiment aflat la limita stîngă a ferestrei, continuînd deplasarea spre stînga, fereastra va baleia tabela, pînă la limita stîngă a acesteia. Aceasta se poate constata observînd modificarea etichetelor literale și numerice ale liniilor și coloanelor, pe laturile stîngă și superioară ale ferestrei. În cazul unui calculator cu o memorie RAM de 48 Ko, se poate manipula o tabelă de dimensiuni apreciabile.

### 3. Introducerea datelor și a textelor.

În principiu VU-CALC se poate examina ca o tabelă inteligentă, în care se pot face diverse planificări, se pot introduce texte sau date numerice.

VU-CALC *acceptă patru tipuri de intrări: text, date, formule și comenzi.*

Pentru a introduce un text se poziționează cursorul în compartimentul la care se dorește plasarea textului, se acționează tasta "și apoi se introduce textul dorit. Pe măsura introducerii textului, acesta va apărea la baza ecranului, avînd cursorul roșu care se va deplasa odată cu introducerea textului. Textul poate fi introdus pe întreaga linie, folosindu-se toate facilitățile de editare. La terminarea textului, se acționează tasta CR, pentru a-l insera în tabelă.

Pentru a introduce un număr într-un compartiment oarecare, se poziționează cursorul în compartimentul dat și se tastează numărul respectiv, după care se acționează tasta CR. Numărul va apărea imediat în compartimentul respectiv.

Pentru a calcula cu ajutorul unei formule valoarea care trebuie plasată într-un compartiment, se poziționează cursorul în compartimentul respectiv și se introduce formula dorită. Cînd formula apare în forma corectă la baza ecranului, se tastează CR. În mod automat formula va fi evaluată și rezultatul obținut se va plasa în compartimentul indicat de cursor. *O formulă se poate aplica în mod automat la mai multe compartimente, folosind comanda Repeat.*

VU-CALC face întotdeauna distincție între text, date și formule.

Cînd linia de la baza ecranului este vidă, tipărind „“, se va aduce VU-CALC în modul comandă, iar lista comenzilor va apărea în primele două linii ale ecranului. Prin tastarea primului caracter dintr-o comandă, aceasta se va executa, pe baza parametrilor dați.

### 4. Utilizarea formulelor

Adevărata forță a programului VU-CALC rezidă în folosirea formulelor, la nivelul de compartimente, linii, coloane, pentru a genera datele necesare actualizării și completării tablei date.

Formulele pot conține constante (numere), referiri la numerele din alte compartimente și operatori aritmetici simpli: +, -, \*, /

Numărul aflat într-un compartiment este referit folosind coordonatele compartimentului: litera liniei/rîndului (r) și numărul coloanei (c)

La alcătuirea formulei, compartimentul se tratează ca o referire la o variabilă, iar formula ca o simplă expresie algebrică ce conține aceste variabile, constante și operatori aritmetici.

Exemple de formule:

$$C1 * 2.05$$

$$C10 * (A2 + 3.3) / B2$$

$$A4 - D4$$



O formulă se poate referi la un compartiment dat sau se poate extinde la o întreagă linie sau coloană. În acest scop se activează comanda #Repeat, care apare în lista de comenzi. Aplicarea formulelor are un caracter relativ. Astfel, în comanda Repeat, dacă formula curentă este extinsă la o secvență de compartimente, în lungul unei linii, atunci, în mod automat, numărul coloanei se incrementează pe măsura ce formula baleiază compartimentele liniei. De exemplu, dacă formula  $1.55 * D1$ , care se aplică la compartimentul D2, se extinde în lungul liniei D, atunci, pentru compartimentul D3, ea va avea aspectul  $1.55 * D2$ , iar pentru D4 va fi:  $1.55 * D3$ . Același lucru se petrece și la aplicarea formulei în lungul unei coloane: eticheta liniei se incrementează corespunzător, secvențial. Dacă în formula dată se dorește referirea la un compartiment absolut, referire ce nu se modifică la repetare, se va folosi prefixul „\$”. Astfel, pentru exemplul de mai sus, formula  $1.55\$ * D1$ , aplicată la linia D, se va referi întotdeauna la compartimentul D1.

## 5. Utilizarea comenzilor.

În VU-CALC există o varietate de comenzi pentru încărcarea, salvarea și tipărirea fișierelor, pentru editare etc. Aceste comenzi sînt introduse prin tastarea simbolului „#”, urmat de primul caracter al comenzii dorite:

- #B            *Blank*: videază compartimentul particular dat.
- #C            *Compute*: forțează recalcularea întregului tabel, în cazurile cînd se modifică o formulă.
- #E            *Edit*: permite modificarea formulei din compartimentul curent sau înlocuirea ei cu altă formulă.
- #F, c, f, j   *Format*: specifică modul de reprezentare al unui număr într-o coloană, în conformitate cu definiția dată prin cei trei parametri: c, f, j.

Parametrul c trebuie să fie un număr zecimal (cu unul sau două ranguri) sau litera „A”. Dacă c este un număr, formatul se va aplica numai la coloana dată; dacă este A, formatul se va extinde la întreaga tabelă.

Parametrul f specifică tipul formatului: I (întreg), \$ (real cu două ranguri zecimale), G (general).

Parametrul j specifică alinierea (justificarea): L (la stînga), R (la dreapta).

- #G, rc        *Go*: deplasează cursorul în compartimentul specificat prin litera liniei r și numărul coloanei c.
- #L            *Load*: șterge ecranul, solicită un nume de fișier și încarcă acel fișier.
- #P            *Print*: tipărește o copie a ecranului la imprimantă
- #Q            *Quit*: anulează tabela curentă de lucru și iese din program.
- #R, rc, f:i   *Repeat*: permite repetarea conținutului compartimentului rc în compartimentele din gama specificată; în acest mod se pot repeta formule. Un compartiment poate fi repetat de-a lungul unei linii sau al unei coloane sau pe un bloc ce cuprinde mai multe compartimente. Blocul se definește cu ajutorul parametrilor gamei f:i, unde f reprezintă referirea la compar-



timentul din colțul din stînga sus, iar  $i$  constituie referirea la compartimentul din dreapta jos, ale blocului. De exemplu, dacă  $f:i$  sînt de forma: A2:B4, compartimentul A2 va fi repetat în compartimentele: A2, A3, A4, B2, B3 și B4.

Compartimentul  $f$  trebuie să fie plasat la stînga sau deasupra compartimentului  $i$ .

#S

*Save:* șterge ecranul și solicită utilizatorului un nume de fișier pentru salvarea pe caseta a tablei.

\$T, r sau c, r

sau c

*Transfer:* linia sau coloana definită de primul parametru este transferată la linia sau coloana definită de cel de-al doilea parametru. O linie nu poate fi copiată într-o coloană sau invers.

## 6. Facilități de sumare

O porțiune dintr-o linie, o coloană sau un bloc dreptunghiular, poate fi adunată, folosind facilitățile oferite de VU-CALC.

Sumarea se execută ca o formulă: se poziționează cursorul în compartimentul în care se dorește plasarea rezultatului, se introduce formula  $\&f:l$  și se acționează tasta CR. Simbolul „&” semnifică suma, iar formula se interpretează ca o sumare a conținutului compartimentelor, de la primul compartiment  $f$ , pînă la ultimul compartiment  $l$ , unde  $f:l$  reprezintă gama blocului de compartimente care se sumează. Astfel,  $f$  specifică compartimentul din colțul din stînga sus al blocului, iar  $l$  — compartimentul din colțul din dreapta jos al blocului.

De exemplu:

$\&B2:C4$

va suma compartimentele B2+B3+B4+C2+C3+C4,

$\&D5:D11$ .

va suma compartimentele liniei D, de la D5, la D11,

$\&A6:C6$

va suma compartimentele coloanei 6, de la linia A, la linia C.

Referirea  $f$  la compartiment trebuie să fie întotdeauna la stînga sau deasupra referirii la compartimentul  $l$ .

## 7. Erori.

La apariția unei erori, datorate introducerii unei formule care se referă la un compartiment vid sau la un compartiment ce conține text și nu date numerice programul va afișa codul erorii în partea de jos a ecranului. În acest caz revenirea în program se face executînd instrucțiunea GO TO 9000. Programul va afișa coordonatele compartimentului asociat cu eroarea. Tastînd CR se va afișa tabela pe care se poate apoi deplasa cursorul în compartimentul în care a apărut eroarea, în vederea corectării acesteia.

## 8. Comenzi VU-CALC — Sumar.

#B

videază compartimentul curent și formula asociată.

#C

forțează recalcularea întregii table în cazul în care s-au modificat data ori formula.

#E

modifică formula în compartimentul curent.

#F, c, f, j

stabilește formatul coloanei  $c$  sau al întregii table ( $A$ );  $f=I$  (întreg),  $f=\$$  (zecimal),  $f=G$  (general); alinierea:  $j=L$  (stînga),  $j=R$  (dreapta).



#G, rc	deplasează cursorul în compartimentul de la linia r, coloana c.
#L	se încarcă fișierul de date VU-CALC.
#P	tipărește o copie a ecranului.
#Q	se iese din program.
#R, rc, f: l	se repetă conținutul compartimentului rc pe gama compartimentelor f:l.
#S	salvează fișierul curent pe casetă.
#T, r sau c, r sau c	transferă linia r, la r sau coloana c, la c sau c

## 14.4. Procesor de texte

Calculatorul HC-85 poate fi utilizat pentru prelucrarea textelor, în condițiile în care este disponibil procesorul de texte Tasword Two.\*

Tastatura calculatorului este folosită, atât pentru introducerea caracterelor alfa-numerice, cât și pentru introducerea comenzilor necesare editării, salvării /încărcării pe/ de pe suport magnetic a fișierului care conține textul în cauză.

*Fișierul text: fereastra; tastatura.*

Procesorul operează pe un fișier text, care conține informația introdusă de la tastatură.

Fișierul text poate avea până la 320 linii, a câte 64 caractere pe linie.

Ecranul TV constituie o fereastră, în cadrul căreia se afișează 22 linii, a câte 64 caractere. Cu ajutorul unor taste de comandă întregul fișier text poate fi deplasat în sus sau în jos, în cadrul ferestrei.

Caracterele alfa-numerice, afișate pe o linie a ferestrei sînt generate prin software și sînt diferite de caracterele afișate în mod normal de calculator. În principiu, în cadrul ferestrei, se pot afișa și numai 32 caractere pe linie, folosind tasta de comanda C, în modul extins, după cum se va explica mai tîrziu.

Caracterele alfa-numerice normale sînt introduse de la tastatura activînd tastele corespunzătoare. Caracterul va apărea pe ecran, în poziția indicată de cursor. Tasta CR (Enter), va deplasa cursorul la începutul noii linii. Pentru afișarea majusculilor, tasta corespunzătoare caracterului dorit, se va activa după ce s-a activat în prealabil tasta CS (CAPS-SHIFT).

Următoarele caractere: (!@£\$%&\* < > ; " - + ? / # , .) se afișează menținînd acționată tasta SS (SYMBOL SHIFT) și acționînd simultan tasta corespunzătoare simbolului dorit.

Caracterele: [ { @ : / } ] pot fi afișate aducînd calculatorul în modul extins (prin activarea simultană a tastelor CS și SS) menținînd apoi activată tasta SS în timp ce se activează tasta corespunzătoare simbolului dorit.

O serie de cuvinte cheie de tipul TO; THEN; <; > etc., indică utilizarea tastelor respective în vederea introducerii unor comenzi.

În cazul în care o tastă este activată mai mult timp, efectul ei se va repeți fie prin introducerea unui caracter, fie prin introducerea unei comenzi.

*Tastele de comenzi.*

Tastele de comenzi devin efective în condițiile în care una din tastele CS sau SS este în prealabil activată, excepție făcînd comanda ENTER, care devine activă la acționarea tastei CR.

\* TASWORD TWD Tasman Software — 1983.



Trebuie menționat faptul că procesorul de texte dispune de două pagini de ajutor, care conțin, sub o formă concisă, descrierea semnificațiilor tastelor de comandă. Aceste pagini pot fi în oricare moment chemate de utilizator, prin activarea simultană a tastelor CS și EDIT (1).

Prima pagină indică semnificațiile tastelor de comenzi în modul normal, în timp ce a doua pagină de comandă furnizează semnificația tastelor de comandă în modul extins.

În continuare se prezintă elementele mai importante din prima pagină de ajutor:

EDIT .....	pagina de ajutor,
CS .....	blocarea tastaturii pe majuscule,
TRUE VIDEO .....	deplasează cursorul peste un cuvânt la stînga,
INV VIDEO .....	deplasează cursorul peste un cuvânt la dreapta,
ARROWS .....	deplasarea cursorului,
GRAPHICS .....	caracterele de comandă pentru imprimantă,
<= .....	deplasează linia la stînga,
<> .....	centrează linia,
>= .....	deplasează linia la dreapta,
AND .....	inserează o linie/un caracter,
OR .....	se merge la sfîrșitul textului,
AT .....	se merge la începutul textului,
STOP .....	încarcă/stochează/tipărește textul,
NOT .....	înlătură o linie,
STEP .....	aduce textul de la linia ce conține cursorul, la sfîrșitul precedent
TO .....	defilare ecran în jos,
THEN .....	defilare ecran în sus,
ENTER .....	se trece la o nouă linie de text,
CS+SS .....	se intră sau se iese din modul extins,
CS .....	se revine în text

Intrarea în cea de-a doua pagină de ajutor se face prin activarea simultană a tastelor CS și SS.

În continuare se prezintă semnificația tastelor de comandă, în modul extins conform informației date în cea de-a doua pagină de ajutor:

Tastele de comandă în modul extins.

#### *Defilare*

F .. defilare rapidă în jos  
G .. defilare rapidă în sus,

#### *Imprimanta ZX*

P .. tipărește fișierul text  
L .. activarea indicatorului de tipărire cu caractere mari,  
K .. dezactivarea indicatorului de tipărire cu caractere mari

#### *Formatare*

E .. activare/dezactivare aliniere la dreapta,  
W .. activare/dezactivare transfer cuvînt în linie nouă, dacă în linia veche nu mai încap,  
J .. aliniere rînd  
K .. nealiniere rînd

#### *Margini*

A .. stabilește limita stînga la poziția cursorului



### *Diverse*

C .. modifică fereastra pe text  
X .. anulează fișierul text  
R .. înlocuiește sau găsește text,  
I .. activează/dezactivează modul de inserție,  
EDIT .. pagină de ajutor,  
ARROWS .. deplasare cursor

S .. aduce limitele la normal  
D .. stabilește limita dreaptă la poziția cursorului

### *Comenzi de bloc*

B .. marchează începutul blocului,  
V .. marchează sfârșitul blocului de text,  
N .. copiază blocul marcat la poziția cursorului  
M .. deplasează blocul marcat la cursor

Pentru a tipări următoarele caractere se activează tasta SS și tasta în dreptul căreia se afla semnul respectiv.

### *Alinierea.*

Ca și în cazul manipulării cuvintelor care depășesc limita stângă a rîndului, procesorul de texte poate asigura în mod automat alinierea rîndurilor la dreapta, prin mărirea corespunzătoare a spațiului dintre cuvinte.

Opțiunea de aliniere automată poate fi blocată, astfel încît, cuvîntul care depășește limita dreaptă a rîndului nu mai este adus automat la începutul rîndului următor.

### *Cursorul înalt.*

Atunci cînd se introduce ultimul caracter din rîndul curent, procesorul de texte deplasează cursorul la începutul următorului rînd al textului. În acest moment cursorul va deveni mai înalt. Dacă în acest moment se introduce un caracter, procesorul presupune că acesta face parte din cuvîntul anterior, de la sfîrșitul rîndului precedent și va aduce întregul cuvînt pe noul rînd. Pentru a evita asemenea situații se va introduce un spațiu.

În cazul în care ultimul caracter de pe un rînd este un semn de punctuație, procesorul de texte va ignora faptul că la începutul noului rînd, cursorul fiind înalt, s-a tastat un caracter.

### *Recomandări privind introducerea textului.*

1. La terminarea unei fraze sau după un semn de punctuație, se va introduce cel puțin un spațiu.

2. La începutul unui nou paragraf se vor introduce unul sau mai multe spații, lăsîndu-se eventual un rînd liber între paragrafe.

### *Încărcarea și execuția procesorului de texte.*

Pentru a încărca procesorul de texte se plasează caseta în dispozitivul de citire și se introduce comanda LOAD " " CR. Încărcarea va avea loc automat în două etape, cu afișarea mesajelor:

program  
bytes

La terminarea încărcării difuzorul va emite semnale sonore cîteva secunde, permițînd oprirea casetofonului.

Procesorul de texte devine activ (după încetarea semnalelor sonore. Cursorul va fi afișat cu intermitență, la începutul unui fișier de text vid. La baza ecranului se va afișa informația referitoare la numărul liniei curente, precum și alte elemente privind opțiunile.



În continuare se poate introduce un nou text sau se poate încărca de pe bandă un fișier text creat anterior, conform indicațiilor date în continuare.

De asemenea, este posibilă încărcarea manualului Tasword (Tasword Tutor), pentru ca utilizatorul să-și poată reîmprospăta elementele necesare utilizării procesorului de texte (a se vedea paragraful respectiv, dat mai jos).

În cazul în care, din diferite motive, s-a intrat în interpretorul de BASIC, se poate reveni în procesor prin activarea comenzii RUN.

### *Salvarea procesorului Tasword.*

În vederea obținerii unei copii a procesorului se activează comanda STOP (SS și A), în timp ce are loc execuția programului. Pe ecran va apare o listă de opțiuni, conform paginilor de ajutor. În continuare se va activa tasta T și se vor urmări indicațiile care vor apărea pe ecran.

### *Salvarea și încărcarea fișierelor text.*

#### *Salvarea:*

Textul introdus în calculator poate fi salvat ca fișier text. În acest scop, în timp ce procesorul de texte este operațional, se va activa comanda STOP (SS și A), ceea ce va avea ca efect afișarea pe ecran a unei liste de opțiuni. Se alege opțiunea Save Text file, activând tasta S, după care procesorul solicită numele care se dorește a fi dat acestui fișier text. Numele poate avea până la 10 caractere. După tastarea numelui fișierului, se activează tasta CR și se vor executa instrucțiunile care apar pe ecran. La terminarea încărcării textului de pe casetă, se poate opta pentru verificarea fișierului înregistrat. În urma unui răspuns afirmativ (Y), se impune derularea casetei la începutul textului și pornirea casetofonului pentru redare.

După terminarea verificării sau dacă această opțiune nu a fost solicitată (N), va apărea o listă de opțiuni, conform efectului comenzii STOP. Pentru revenirea în fișierul text, se activează Y.

Dacă, la verificare, a apărut un mesaj de eroare („Tape Loading Error“), se ajunge automat în interpretorul de BASIC. Pentru a reveni în procesorul de texte se activează comanda RUN. Se vor repeta operațiile de salvare și verificare. În cazul apariției unei noi erori se va schimba caseta magnetică.

#### *Încărcarea.*

La încărcarea unui fișier text, automat se va distruge textul fișierului curent.

Încărcarea se face executând mai întâi comanda STOP, în timp ce procesorul de texte operează. Se va activa tasta J, pentru a selecta opțiunea de încărcare a fișierului text („Load Text File“). Se va cere introducerea numelui fișierului text, care urmează a se încărca. Se tastează numele fișierului, se activează CR și se pornește casetofonul. Dacă nu se introduce un nume de fișier text, se va încărca automat primul fișier text întâlnit pe casetă. Numele fișierului text va apărea pe ecran, imediat după încărcare.

#### *Fuzionarea textelor.*

Fuzionarea constă în încărcarea unui fișier text de pe casetă și plasarea lui în continuarea fișierului text curent al procesorului. În acest scop, după execu-



ția comenzii STOP se va activa tasta M (Merge — fuzionare), pentru a selecta opțiunea de fuzionare. Pe ecran vor apărea indicații asemănătoare celor de la încărcare, date mai sus.

Fuzionarea efectivă nu va avea loc, dacă în urma acestei operații se depășește numărul de 320 de linii de text. Automat se va intra în interpretorul BASIC. Pentru a lansa procesorul de texte se va activa comanda RUN.

### *Manualul pentru Tasword.*

Manualul pentru Tasword (Tasword Tutor) reprezintă un fișier text care permite utilizatorului să se familiarizeze cu tastele de comandă.

În principiu manualul de Tasword este înregistrat imediat după oțetții „bytes” ai procesorului de texte.

După încărcarea procesorului de texte se va proceda ca la încărcarea unui fișier text oarecare:

1. Se activează comanda STOP (SS și A).
  2. Se selectează opțiunea de încărcare a fișierului text („load text file”), activînd tasta J.
  3. Se activează tasta CR (ENTER).
  4. Procesorul de texte va solicita numele fișierului text. Se va activa tasta CR (ENTER), pentru a lansa citirea primului fișier text de pe casetă.
  5. Se pornește casetofonul.
- Manualul (Tasword Tutor) va apărea pe ecran imediat după terminarea încărcării. Se oprește casetofonul.

### *Tastele de comandă.*

În continuare se vor prezenta tastele de comandă în modul normal.

Procesorul de texte se află în modul normal, cînd linia de la baza ecranului nu este afișată cu intermitență.

Cînd tasta de comandă este activată, este necesar ca și tasta CS sau SS să fie deja activată. Excepție face comanda ENTER.

#### **EDIT (CS și 1)**

În mod normal se afișează pagina de ajutor. Pentru a trece la pagina următoare de ajutor (modul extins), se activează CS și SS iar pentru a reveni în text se activează CR.

#### **CAPS LOCK (CS și 2).**

Menținînd activată tasta CS, se pot introduce majuscule. După introducerea comenzii CAPS LOCK, toate literele tastate vor fi tratate ca majuscule. Această situație este semnalizată la baza ecranului.

#### **TRUE VIDEO (CS și 3).**

Această comandă deplasează cursorul la sfîrșitul cuvîntului aflat la stînga cursorului. Astfel, cursorul poate fi deplasat rapid în lungul unei linii.

#### **INV VIDEO (CS și 4).**

Această comandă deplasează cursorul la sfîrșitul următorului cuvînt, aflat la dreapta cursorului, putîndu-se parcurge rapid o linie.

Dacă la dreapta cuvîntului nu se mai află text, pe o scurtă durată de timp, procesorul va face o verificare în acest sens.

#### **ARROWS (CS și 5, 6, 7, 8).**

Săgețile 5, 6, 7 și 8 sînt folosite pentru a deplasa rapid cursorul în orice poziție pe ecran.



#### GRAPHICS (CS și 9).

Aceste taste sînt folosite pentru tipărirea simbolurilor grafice asociate tastelor 1—8. Modul grafic este semnalizat printr-un mesaj la baza ecranului.

Pentru a obține caracterele grafice corespunzătoare, simultan va fi activată și tasta CS.

Pentru a ieși din modul grafic se vor activa, din nou, CS și 9.

Caracterele grafice sînt tipărite ca la imprimanta ZX, dar pot fi folosite și în calitate de caractere de comandă pentru imprimantele normale.

#### DELETE (CS și 0).

Comanda DELETE înlătură caracterul din dreptul cursorului și deplasează restul liniei aflate la dreapta cursorului, cu o poziție spre stînga.

Caracterele omise pot fi plasate peste alte caractere, conform poziției curente a cursorului, fără a fi necesară înlăturarea vechilor caractere, prin semnalul DELETE.

#### <= (SS și Q).

Comanda deplasează textul sub cursor și la stînga lui cu o poziție. Ea nu va avea efect, dacă există deja un caracter la marginea stîngă. Textul aflat între limite nu este afectat de această comandă, cînd cursorul se află între limite.

#### < > (SS și W).

Această comandă centrează textul pe linia ce conține cursorul, între limite. Ea este utilă pentru titluri. Textul aflat între limite nu este deplasat, cînd cursorul se află între limite.

#### >= (SS și E).

Această comandă deplasează textul sub cursor și la dreapta lui cu o poziție. Ea nu are efect, dacă caracterul se află la limita dreaptă. Textul aflat între limite nu este deplasat, neavînd loc nici o acțiune, dacă cursorul se află în interiorul limitelor.

#### AND (SS și Y).

Comanda este folosită pentru a insera noi linii, cuvinte și caractere, în fișierul text.

Pentru a insera un rînd vid, se poziționează cursorul la începutul liniei, care urmează liniei ce trebuie inserate. Se activează AND, pentru inserarea unei noi linii.

Pentru a insera cuvinte suplimentare, între cuvintele deja existente într-un text, se poziționează cursorul în spațiul dintre cuvinte. Activînd comanda AND, cuvintele aflate la dreapta cursorului sînt deplasate pe o nouă linie. Astfel, se poate introduce textul adițional. Pentru a crea un rînd liber, în continuare, se activează tasta AND, conform „Modului de inserție“, descris mai jos.

Pentru a insera un caracter în mijlocul unui cuvînt, se poziționează cursorul pe caracter, la dreapta poziției cerute.

La activarea comenzii AND, rîndul nu este aliniat, creîndu-se un spațiu pentru noul caracter. Dacă rîndul nu poate fi aliniat, atunci se va crea o nouă linie, după cum s-a arătat în paragraful precedent.

Aceste proceduri de inserție vor distruge alinierea paragrafului. Alinierea se poate reface folosind comanda STEP.

Comanda AND nu are efect în interiorul limitelor, cu excepția cazului cînd cursorul este în coloana 1.

#### OR (SS și U).

Comanda duce la găsirea și afișarea sfîrșitului fișierului text.

#### AT (SS și I).



Această comandă asigură trecerea la începutul fișierului text.

STOP (SS și A).

Comanda este folosită pentru salvarea, încărcarea și tipărirea fișierului text. După activarea acestei comenzi se afișează următoarele opțiuni:

- |  |          |
|--|----------|
| — tipărește fișierul text (print text file)              | <i>p</i> |
| — salvează fișierul text (save text file)                | <i>s</i> |
| — încarcă fișierul text (load text file)                 | <i>j</i> |
| — fuzionare text (merge text file)                       | <i>m</i> |
| — revenire la fișierul text (return to text file)        | <i>y</i> |
| — definește grafica/imprimanta (define graphics/printer) | <i>g</i> |
| — salvează procesorul de texte (save Tasword)            | <i>t</i> |
| — revenire în BASIC (into BASIC)                         | <i>b</i> |

Activînd *y* se trece în fișierul text. Activînd *b* se trece în BASIC, ceea ce permite activarea altor programe, în timp ce procesorul de texte este în memoria calculatorului.

Lansarea procesorului de texte se face cu comanda RUN.

NOT (SS și S).

Comanda înlătura linia pe care se afla cursorul. Toate liniile aflate mai jos de cursor se deplasează cu o poziție mai sus.

STEP (SS și S).

Această comandă reformatează textul de la linia ce conține cursorul, pînă la sfîrșitul paragrafului. Sfîrșitul paragrafului este detectat de un spațiu liber sau de un semn de punctuație.

Comanda STEP este utilă pentru organizarea corectă și estetică a unui text, în care s-au efectuat inserții sau extracții. Dacă opțiunea de aliniere este activă, numai textul dintre limite va fi reformatat.

Comanda STEP va realinia textul, dacă alinierea la dreapta este activă și-l va lăsa nealiniat, dacă nu este activă.

TO (SS și F).

Comanda TO asigură defilarea ecranului în jos cu o linie din fișierul text.

THEN (SS și G).

Comanda THEN asigură defilarea ecranului în sus cu o linie din fișierul text.

ENTER (CR).

Această comandă deplasează cursorul la începutul unei noi linii.

În „Modul de inserție” se inserează o nouă linie.

### *Tastele de comandă în modul extins.*

Pentru a intra sau ieși în/din modul extins se acționează simultan tastele CS și SS. Starea corespunzătoare modului extins este semnalizată prin afișarea cu intermitență a liniei de la baza ecranului. În modul extins pot fi introduse numai următoarele caractere: [ ] @ : \ [ ], prin activarea tastei SS și a tastei asociate caracterului respectiv.

În modul extins se utilizează următoarele comenzi:

EDIT (CS și I).

Această comandă aduce pe ecran pagina de ajutor, corespunzătoare modului extins, cînd procesorul se află în modul extins. Pentru a se afișa pagina de ajutor, cînd procesorul se află în modul normal, se activează simultan tastele CS și SS. Revenirea în modul normal se realizează activînd tasta CR.



**ARROWS (CS și 5, 6, 7, 8).**

Cu ajutorul acestor comenzi se deplasează cursorul ca și în modul normal.

**W** — activare/dezactivare a poziționării automate, la începutul liniei următoare, a cuvântului curent când acesta nu mai încapă în linia curentă.

Mesajul „W/W“, care apare la baza ecranului, indică activarea acestei facilități.

**E** — activare/dezactivare aliniere automată la dreapta.

Mesajul „R justify“, de la baza ecranului, specifică activarea acestei facilități. În acest caz textul este automat aliniat la dreapta, prin introducerea unor spații libere între cuvinte.

**R** — înlocuiește sau găsește textul.

Această facilitate permite găsirea primei apariții a unui cuvânt dat sau înlocuirea tuturor aparițiilor unui cuvânt dat, începînd de la poziția curentă a cursorului, cu un alt cuvânt sau un grup de cuvinte. Pentru a activa această operație, de la începutul textului, se folosește tasta de comandă **AT** în modul normal, după care, în modul extins se utilizează comanda **R**.

În modul extins, după activarea comenzii **R**, procesorul de texte va solicita cuvîntul care trebuie căutat sau înlocuit. În continuare, trebuie introdus cuvîntul respectiv, fără a se folosi spații la începutul său. Activarea tastei **CR**, va indica procesorului terminarea introducerii cuvîntului. După aceasta procesorul va solicita textul care va înlocui cuvîntul respectiv. Pentru a găsi următoarea apariție a cuvîntului, se va activa tasta **CR**. În vederea înlocuirii următoarelor apariții ale cuvîntului se poate introduce un text de pînă la 32 caractere, incluzînd și spații, după care se activează **CR**.

Procesorul va reformata fiecare paragraf ce include cuvîntul dat, în conformitate cu starea de activare/dezactivare a facilității de aliniere la dreapta, la momentul respectiv.

**I** — activare/dezactivare a modului de inserție.

Cînd această comandă este activată, procesorul creează o nouă linie vidă, pentru a fi folosită atunci cînd linia curentă de text este completă sau cînd s-a activat **CR**.

Facilitatea este utilă cînd se dorește inserția unor linii de text în mijlocul unui alt text, deja introdus.

**P** — tipărește textul la imprimantă

În cazul în care, la microcalculator este conectată o imprimantă **ZX**, fișierul text va fi tipărit la imprimantă, după activarea tastei **P**. Pe ecran va defila textul, pe măsură ce se va tipări la imprimantă. Pentru blocarea operației se activează tasta **Q**.

**A** — stabilește limita stîngă.

La activarea acestei comenzi, limita stîngă este fixată imediat înaintea poziției curente a cursorului. Stabilirea limitei este indicată prin schimbarea culorii unei părți din ecran.

**S** — anulează limitele stabilite.

La activarea acestei comenzi se anulează limitele stabilite, revenindu-se la limitele normale: extremele stîngă și dreaptă ale ecranului.

**D** — stabilește limita dreaptă.

Această comandă stabilește limita dreaptă imediat după poziția curentă a cursorului. Poziția limitei din dreapta este marcată prin schimbarea culorii ecranului.



F — defilare rapidă în jos.

La activarea acestei comenzi, în modul extins, are loc defilarea imaginii de pe ecran cu 22 de linii în jos.

G — defilarea rapidă în sus.

La activarea acestei comenzi, în modul extins, ecranul defilează cu 22 de linii în sus.

J — limitarea liniei.

Prin activarea acestei comenzi, linia pe care se află cursorul este limitată la dreapta.

H — linie nelimitată.

La activarea acestei comenzi, linia pe care se află cursorul nu mai este limitată la dreapta, prin înlăturarea spațiilor suplimentare între cuvinte.

L — activarea indicatorului de tipărire a literelor de înălțime dublă, la imprimanta ZX.

La activarea acestei comenzi, în modul extins, se inserează în text, deasupra liniei pe care se află cursorul, mesajul: „print at double height on“ (activare tipărire la înălțime dublă). Activarea comenzii trebuie făcută atunci când cursorul se află la cap de linie. Indicatorul nu va fi tipărit la imprimantă, dar textul care urmează va avea înălțimea dublată, față de situația normală. Pentru a anula acest indicator, se va folosi tasta de comandă NOT, din modul normal.

K — dezactivarea indicatorului de tipărire cu litere de înălțime dublă, la imprimanta ZX.

La activarea acestei comenzi, se inserează deasupra liniei pe care se află cursorul mesajul: „print at double height off“ (anularea indicatorului de tipărire cu litere de înălțime dublă). Anularea indicatorului se realizează prin activarea tastei de comandă NOT, în modul normal.

X — anulează fișierul text.

La activarea acestei comenzi, în modul extins, întregul fișier este înlăturat, din fișierul text. Pentru a preveni înlăturarea accidentală a textului, procesorul de texte solicită o confirmare suplimentară.

C — modifică Fereastra pe Text.

În modul extins, această comandă este folosită pentru a deschide sau închide o fereastră de 32 caractere în text. Când Fereastra este deschisă, bordura ecranului își schimbă culoarea, pentru a indica acest lucru.

Dacă Fereastra a fost deschisă, textul va fi afișat pe ecran cu 32 caractere pe linie, de dimensiuni obișnuite. Fereastra poate defila lateral, folosind tastele marcate cu săgeți, care deplasează și cursorul. Defilarea laterală are loc automat, pe măsură ce se introduce textul.

B — indicatorul de început de bloc.

Blocurile de text pot fi deplasate sau copiate dintr-o zonă în alta ale fișierului. Înainte de deplasare sau copiere, începutul blocului de text trebuie marcat. În acest scop se activează tasta B, în modul extins. Marcarea pe ecran se va face printr-o paranteză mare deschisă, pe linia aflată deasupra primei linii a blocului.

După introducerea indicatorului, procesorul va verifica dacă anterior a mai fost introdus un asemenea indicator, operația va necesita un anumit interval de timp.

Pentru anularea indicatorului de început de bloc de text, se deplasează cursorul la începutul liniei ce conține indicatorul și se activează comanda NOT, în modul normal.



V — indicator de sfârșit de bloc de text.

Această comandă este folosită pentru a indica linia pe care se află cursorul ca fiind ultima linie a blocului de text. Procesorul va introduce o paranteză mare închisă sub ultima linie a blocului.

În prealabil procesorul va verifica dacă blocul de text nu a fost deja închis.

Pentru a înlătura indicatorul de bloc de text se activează tasta NOT, în modul normal.

M — deplasează blocul de text.

Un bloc de text marcat va fi deplasat la o nouă poziție, atunci când tasta M este activată, în modul extins. Textul este deplasat la noile linii create, deasupra liniei ce conține cursorul, în momentul activării comenzii M.

N — copiază un bloc de text.

Acțiunea acestei comenzi, în modul extins, are ca efect copierea blocului de text deasupra liniei pe care se află cursorul.

*Observații privind stabilirea limitelor textului.*

Tastele de comandă A și D, în modul extins, sînt folosite pentru stabilirea limitelor stînga și dreapta ale textului ce urmează a fi introdus.

Tasta de comandă S, în modul extins, anulează limitele impuse textului.

După stabilirea limitelor textului, acesta este introdus în mod normal, între cele două limite, cu aliniere automată la nivelul fiecărui rînd, ca și cînd cele două limite stabilite ar reprezenta limitele ecranului.

Folosirea limitelor este utilă pentru a putea evidenția rapid anumite paragrafe sau zone din text.

Stabilirea limitelor este marcată prin schimbarea culorii ecranului pe anumite zone date.

Tastele de comandă, marcate cu săgeți, pot fi utilizate pentru deplasarea cursorului, între limitele stabilite pentru text, în vederea plasării unor noi limite sau pentru a poziționa un text între limite date.

Tastele de deplasare  $\Leftarrow$ ,  $\Rightarrow$  și centrare  $\langle \rangle$  nu afectează textul aflat între limite și nu operează cînd cursorul este pe una din limite.

Tasta AND, de inserție a unui text, nu operează între limite, cu excepția cazului cînd cursorul se află în prima coloană.

Tasta de comandă STEP reformează numai textul aflat între limita stîngă și cea dreaptă.

Comanda de găsire și înlocuire, R — în modul extins, ignoră limitele stabilite.

Realinierea automată a paragrafului, care are loc la înlocuirea unui text, poate modifica formatul unui text, care a fost introdus între limitele date.

*Tipărirea textului la imprimantă.*

Microcalculatorul HC-85 posedă interfața serială la care se poate conecta o imprimantă serială standard, de exemplu imprimanta SCAMP 9335, operînd la o viteză de 9600 bauds.

Pentru tipărirea unui text se folosește comanda STOP, în modul normal, care oferă mai multe opțiuni, printre care și pe aceea de tipărire a fișierului text.

Urmărind indicațiile care apar pe ecran, se poate tipări fișierul text.

Folosind imprimanta 9335 în modul grafic se poate tipări conținutul ecranului, cu caracterele special generate, prin software, pentru procesorul de text.



## CALCULATORUL PERSONAL ÎN ÎNVĂȚĂMÎNT ȘI EDUCAȚIE

### Capitolul 15. | Calculatorul personal în procesul de învățămînt

#### 15.1. Informatica în programa învățămîntului liceal

Informatica și calculatorul au pătruns în toate domeniile permeabile algoritmicizării. Învățămîntul, acumulînd pînă în prezent cel mai mare număr de algoritmi, precum și o bogată tehnică de calcul și de reprezentare, solicită aportul acestora pentru optimizarea procesului de predare-învățare, pentru dezvoltarea caracterului său practic-aplicativ, pentru integrarea cu producția și cercetarea. Această solicitare intensivă are loc în contextul noii revoluții tehnico-științifice contemporane și al noii revoluții agrare din patria noastră.

Folosirea calculatorului în procesul de învățămînt a început, timid, cu aproximativ douăzeci de ani în urmă, cînd dezvoltarea limbajelor de programare a făcut posibilă scrierea de programe logice. Cinci ani mai tîrziu s-a reușit cuplarea mijloacelor audiovizuale cu informatica — textul, sunetul și imaginea permițînd de această dată apariția programelor didactice\*, sub denumirea de *programe informatice educaționale*.

\* Preocupări pentru utilizarea informaticii și calculatorului în procesul de predare-învățare din licee apar în perioada respectivă și în țara noastră. În lucrarea „Laboratorul de matematică” Editura Didactică și Pedagogică București — 1973 (autori: Acad. N. Teodorescu, Prof. Ch. N. Rîzescu, Asist. B. Ionescu, Prof. D. Ogrzeanu), capitolul VI conține modele de lecții privind folosirea calculatorului FELIX CE-33 în predarea matematicii. În lucrarea „Laboratorul de matematică — teme și fișe experimentale”, tipărită de I.C.I.E., Buc. — 1978, (autor Prof. Gh. N. Rîzescu), sînt prezentate modele de teme, lecții și fișe de laborator pentru predarea matematicii (pag. 43—136 din Cap. IV). Lucrarea „Totul despre... calculatorul personal aMIC” Editura Tehnică, Buc. — 1985 — coordonator Prof. dr. ing. A. Petrescu, pune bazele folosirii calculatorului în procesul de învățămînt. Lista unor astfel de lucrări continuă.



Unele țări puternic industrializate, ca și unele cu dezvoltare medie sau în curs de dezvoltare, au înscris informatica în planurile de învățământ preuniversitar. În țara noastră, informatica și calculatorul au devenit obiect de studiu în cadrul liceelor de matematică și fizică. Noțiuni de informatică apar și în unele manuale de tehnologie ale liceelor cu profil electrotehnic sau electronic, în programa unor școli profesionale, precum și a altor forme de pregătire și perfecționare a forței de muncă. De asemenea, practica și studiul unor noțiuni de informatică apar și în programa orientativă a activităților școlare facultative privind cercurile de matematică și informatică ale elevilor din gimnazii și licee, aceste activități fiind organizate sub îndrumarea M.E.I., a C.N.O.P. (vezi cap. 16) și U.T.C. O pondere însemnată în orientarea școlară și profesională pentru informatică a tineretului școlar o au și activitățile organizate de Casele Pionierilor sau de Casa Științei și Tehnicii pentru tineret. Fiind puternic solicitate și impulsionate de cerințele dezvoltării noastre economice intensive, cu ajutorul celor mai avansate tehnologii, noțiunile de informatică se implementează generalizat programelor de matematică ale tuturor liceelor țării începând cu anul școlar 1987—1988.\*\*)

Această implementare, unitar și organic realizată, conferă programei o mai mare deschidere intra și interdisciplinară, precum și un puternic caracter formativ și practic-aplicativ.

Însușirea noțiunilor informatice de *algoritm*, *schemă logică* și *pseudocod* (în clasa a IX-a), transcrierea acestora în *limbajul de programare „BASIC”* și utilizarea *microcalculatorului personal* (în clasa a X-a), precum și deprinderea utilizării facilităților grafice ale calculatorului prin aplicațiile prevăzute în clasele a XI-a și a XII-a (respectiv a XIII-a seral) — constituie elemente de bază în perspectiva înțelegerii complexelor probleme ale automatizării, cibernetizării și robotizării, pentru care trebuie pregătită forța de muncă.

Pentru o mai clară evidențiere a uneia din laturile calitative, noi, aduse prin îmbunătățirea programei de matematică a învățământului liceal, zi și seară, vor fi reproduse, în continuare, câteva din motivațiile, aprecierile și recomandările metodologice din programa M.E.I., reeditată în 1987. Este vorba aici, în deosebi, de Capitolul II — privind prioritățile programei, intitulat: „Cu privire la predarea-învățarea informaticii în învățământul liceal”, capitol pe care-l cităm în întregime:

„Predarea noțiunilor de informatică în învățământul liceal asigură formarea unei culturi informatice de bază, însușirea principiilor de programare și utilizare a calculatorului electronic, a deprinderilor și priceperilor necesare utilizării echipamentelor electronice de calcul, aplicarea metodelor și tehnicilor proprii informaticii la predarea celorlalte discipline fundamentale de cultură generală, disciplinelor tehnologice pentru diferite meserii în care se pregătesc elevii. Se va accentua rolul tehnicii de calcul în accesibilizarea informaticii și, mai ales în activitatea creatoare, având în vedere cibernetizarea, robotizarea etc.

Noțiunile de informatică se vor introduce în mod gradat, într-o înlănțuire logică cu cele de matematică, care să permită o însușire corespunzătoare a cunoștințelor. Ele sînt corelate cu noțiunile de matematică introduse în programă. Prin predarea-învățarea lor se va realiza ridicarea calității pregătirii elevilor în domenii cît mai variate.

\* V. și 15.6

\*\* Statuarea Informaticii în cadrul disciplinelor fundamentale ale învățământului preuniversitar este într-o continuă evoluție.



1. LA CLASA A IX-A se introduce noțiunile legate de algoritmi și reprezentarea lor; în clasa a X-a, elemente de programare în limbajul „BASIC”, urmînd ca în clasele a XI-a și a XII-a să se realizeze un număr de aplicații în matematică, fizică, chimie etc., cu ajutorul microcalculatorului personal.

Noțiunile despre algoritmi, care constituie un capitol separat al programei, sînt necesare rezolvării problemelor cu ajutorul calculatorului, găsirea algoritmului de calcul constituind elementul fundamental în rezolvarea unei probleme. Se va prezenta un scurt istoric al noțiunii de algoritm cu exemple clasice (algoritmii operațiilor aritmetice, algoritmul lui Euclid etc.), urmat de reprezentarea algoritmilor prin scheme logice și exemple.

2. LA CLASELE A X-A capitolul cuprinde elemente de prelucrare automată a datelor. Se vor prezenta cunoștințele referitoare la structura și funcționarea microcalculatoarelor personale și elemente de programare în limbajul BASIC, cu excepția facilităților grafice ale limbajului care vor fi predate în clasa a XI-a.

3. LA CLASELE A XI-A și A XII-A se vor rezolva o serie de probleme din matematică prin programarea lor în limbajul BASIC.

Aplicațiile menționate în programă vor fi prezentate în noua viziune determinată de procedee și metode informatice (elaborarea algoritmului și a programului), evitînd astfel prezentarea obișnuită (clasică).

Noțiunile se vor introduce eșalonat: pentru clasa a IX-a, în anul școlar 1987—1988; pentru clasa a X-a, în anul școlar 1988—1989, pentru clasa a XI-a în anul școlar 1989—1990, iar pentru clasa a XII-a în anul școlar 1990—1991.

4. ÎN CADRUL CERCURILOR ORGANIZATE ÎN ȘCOLI se pot realiza activități complexe cu elevii, precum și inițierea acestora în alte limbafe de programare.

Se poate sugera, în cadrul cercurilor de matematică, să se extindă sfera preocupărilor teoretice și practice pentru realizarea unor algoritmi privind:

- |  |  |
|--|--|
| — trasarea graficului funcției trinom de gradul II și rezolvarea inecuațiilor de gradul I sau gradul II;         | — reprezentarea grafică a funcției exponențiale și a funcției logaritmice; |
| — rezolvarea grafică a sistemelor de două ecuații de gradul II cu două necunoscute și a sistemelor de inecuații; | — compunerea funcțiilor;   |
| — trasarea graficului funcției putere;   | — secțiuni în corpuri geometrice;  |
| — intersecția cercului cu dreapta;   | — aplicații cu matrice;  |
| — trasarea liniilor importante în triunghi și rezolvarea triunghiului;   | — rezolvarea sistemelor de tip $(n,n)$ ;                                   |
| — reprezentarea grafică a funcțiilor trigonometrice;   | — convergența șirurilor;   |
|  | — aplicații ale calculului vectorial;                                      |
|  | — probleme de programare liniară;  |
|  | — locuri geometrice tratate analitic;                                      |
|  | — reprezentarea conicelor ca secțiuni;                                     |
|  | — probabilități geometrice;  |
|  | — probleme de statistică.  |

În acest mod, muncitorul de mîine va poseda o cultură informatică de bază pe care o va folosi în activitatea de informare, de prelucrare și utilizare creatoare a informației la locul său de muncă. Tehnica informatică îi va fi necesară în culegerea și prelucrarea rapidă a datelor rezultate din procesul muncii, pentru adoptarea operativă a deciziei optime în conducerea și executarea proceselor tehnologice, în perfecționarea lor prin munca de cercetare. Se poate spune că, prin îmbunătățirile aduse actualei programe de matematică a liceului, informatica și calculatorul își ocupă locul ce li se cuvine atît ca „obiect” de învățămînt, cît și ca „mijloc” de învățămînt.



Acesta este un pas hotărîtor pe linia perfecționării învățămîntului, a integrării lui tot mai strîns cu cercetarea științifică și cu producția, cu modernizarea accelerată a întregii vieți economice și sociale.

Desigur, este necesar să-i obișnuim pe elevi să aplice noțiunile informatice de algoritm, schemă logică și program în cadrul altor discipline școlare, inclusiv pentru pregătirea lor în meserie. De asemenea, se recomandă folosirea informaticii și calculatorului în cadrul unor lucrări de diplomă, al lucrărilor de cercetare prezentate la sesiuni de referate și comunicări științifice, sau chiar în cadrul problemelor de evidență din școală.

Experiența școlară de pînă în prezent a validat ideea că algoritimizarea și prezentarea grafică a informației sînt foarte accesibile recepționării și prelucrării de către cel care învață. Este demn de evidențiat faptul că echipamentele electronice de calcul pentru procesul de învățămînt s-au dezvoltat și ieftinit permanent, devenind treptat accesibile învățămîntului preuniversitar. De asemenea, au apărut și se perfecționează o serie de limbaje de programare ușor accesibile tinerei generații, cum ar fi: „BASIC-ul“, „LOGO“,....

Devine deci necesară și posibilă înzestrarea laboratoarelor școlare cu tehnică electronică pentru asistarea procesului de predare-învățare, pentru desfășurarea activității cercurilor școlare pe obiecte, sau interdisciplinare. În plus, apariția conceptului de „informație“ și integrarea sa în programul școlii are menirea de a dezvolta concepția materialist-științifică asupra materialității și cognoscibilității lumii, asupra posibilităților teoriei instruirii și educației ca procese informaționale dezvoltate prin practica școlii și a societății.

Dar ce înseamnă „asistarea cu calculatorul a procesului de învățămînt“?

Mai întîi, cîteva chestiuni generale.

Învățarea cu calculatorul se bazează pe dialogul dintre mașină și cel care învață, ceea ce conferă procesului învățării un **caracter interactiv**. Acest dialog este conceput pe baza unui program didactic și se realizează cu ajutorul unei console, legată de calculator-alcătuită dintr-o claviatură alfanumerică și dintr-un ecran. Lecția este stocată în memoria calculatorului. Cu ajutorul software-ului, ea este afișată secvențial, sub forma unor imagini-ecran. Prin intermediul claviaturii se asigură dialogul cu mașina, cel care învață răspunzînd la întrebările primite. Caracterele marcate de el pe claviatură apar pe ecran și trimit la program. Mașina analizează răspunsul cu ajutorul programului și, în funcție de această analiză, apare pe ecran un nou text conținînd noi întrebări (sau îndrumări suplimentare) pentru utilizator.

În cazul calculatorului individual, pe care-l prezentăm aici ca mijloc de învățămînt, sistemul informatic de pe masa de lucru a elevului este prezentat în „Figura 15.1“ și se compune din:

- calculatorul HC-85 (sau alt tip de calculator „personal“);
- televizorul (sau monitorul TV);
- sursa S (alimentatorul);
- casetofonul C.

Acest sistem informatic este denumit „**trusă informatică**“.



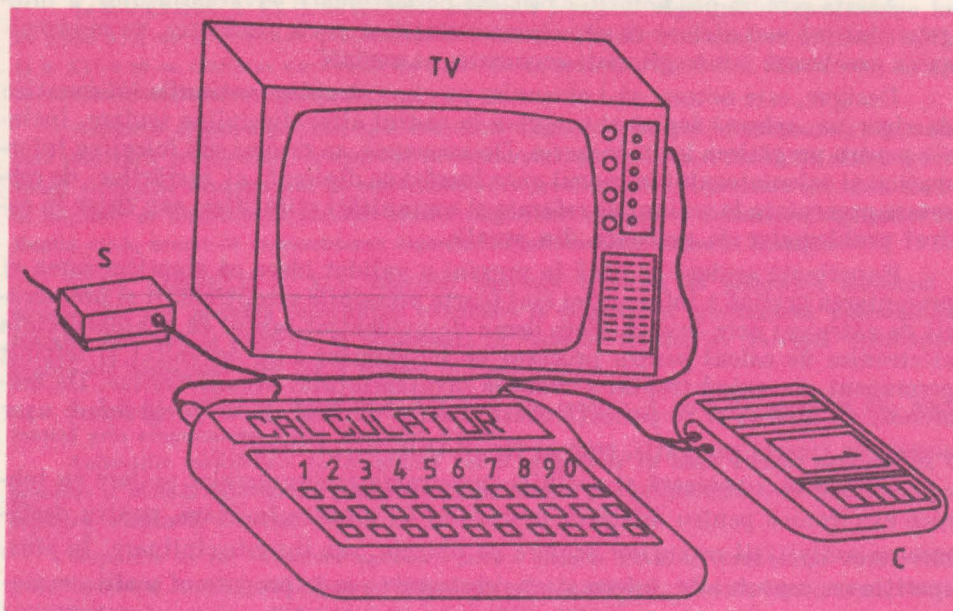


Fig 15.1. Trusa informatică

## 15.2. Asistarea cu calculatorul a procesului de învățămînt

Conceptul de asistare cu calculatorul a procesului de învățămînt constă în folosirea acestuia într-una sau mai multe din situațiile semnalate în cele ce urmează:

1°. Predarea unei lecții, a unui grup de lecții, sau prezentarea unor secvențe dintr-o lecție, în scopul optimizării procesului argumentării, al evidențierii mișcării cu ajutorul desenului de animație, al obținerii imediate a unor rezultate sau efecte așteptate, al accelerării procesului învățării și al dezvoltării spiritului aplicativ.

2°. Verificarea cu calculatorul a unei lecții, sau a unor secvențe din lecție, a unui capitol, a unei discipline de învățămînt, sau chiar a unei programe școlare — în cazul examenelor.

3°. Învățarea cu calculatorul — pe baza unor programe informatice educaționale adecvate învățămîntului autonom, efectuarea de exerciții pentru învățarea unor operații și formarea deprinderilor adecvate.

4°. Dezvoltarea și accelerarea muncii de creație, folosind informatica și calculatorul — prin intermediul cercurilor școlare pe obiecte, al cercurilor integrate și al practicii tehnico-productive.

Evident, calculatorul poate fi folosit și pentru gestiunea școlară.

Trebuie reținut că o mare parte din programele didactice folosite pe calculator sînt prevăzute cu exerciții de control, răspunsurile fiind uneori alese dintr-o listă de variante, afișată pe terminal. În cazul unor răspunsuri eronate, cel ce învață este dirijat (prin așa-zisa funcție de „ajutor“ a calculatorului) să reia



studiul, pe baza unor subprograme, a unor întrebări suplimentare sau aplicații mai detaliate.

**Citeva aspecte pedagogice privind folosirea calculatorului în procesul de învățămînt.**

1°. Calculatorul se poate folosi, în general, ca auxiliar al procesului de învățămînt, la toate disciplinele școlare care solicită tehnica de calcul sau de reprezentare, începînd cu vîrste școlare destul de mici. Pentru cei mici calculatorul poate fi folosit în jocurile cu caracter strategic, pentru desen, citit și scris. În jocurile didactice mișcarea, culoarea și sunetul sînt elemente determinante ale interacțiunii jucător-mașină. În lupta pentru marcarea unui număr cît mai mare de puncte, jucătorul își dezvoltă atenția încercînd diverse strategii pentru depășirea fazelor critice care apar în joc. Astfel începe formarea deprinderilor programatice ale gîndirii la copii, această preocupare putînd fi așezată astăzi pe același plan cu necesitatea dezvoltării reprezentărilor numerice și spațiale. Aici se formează și priceperea de a schematiza și abstractiza, element strict necesar pentru studiul tuturor disciplinelor școlare. În școala primară, calculatorul se poate folosi și pentru îmbunătățirea scrisului unor copii — ale căror deprinderi se instalează mai greu. Exercițiile de scriere asistate de calculator mențin mereu trează atenția elevului pentru a da literei forma sa completă, solicitînd gîndirea și precizia, avertizînd eroarea și localizînd greșeala. Dacă elevul îndepărtează virful „tocului“ cu care scrie pe ecran de la modelul foarte fin și cu întrerupere al literei date, atunci se aprinde un bec, indicîndu-i-se locul unde ar trebui să fie virful „tocului“.

Învățarea cu calculatorul are un demers activ, sprijinit pe caracterul interactiv al dialogului om-mașină. Elevul învață singur și sigur, fără emoții; inhibiția, care apare în procesul „ascultării“ tradiționale (în fața tablei și a clasei), dispăre. Munca cu calculatorul fixează atenția, solicită concentrarea și beneficiază de funcția de ajutor a acestuia la momentul oportun. Elevul caută adeseori „mașina“ în scopul însușirii depline a cunoștințelor.

2°. Calculatorul permite o mai bună intuire a fenomenului studiat, pe baza repetării acestuia pînă la înțelegerea sa deplină. În mod practic, variînd parametrii de care depinde realizarea unei experiențe, se poate realiza studiul acesteia într-o mare varietate de exemplificări, adîncindu-i conținutul și lărgindu-i sfera, sau extrapolînd rezultatele (obținute) dincolo de obiectul predat. Este știut că în învățămîntul tradițional o experiență nu poate fi repetată în timpul predării — datorită constrîngerilor de timp și de materiale. Adeseori, în predarea tradițională a fizicii, chimiei, sau a obiectelor tehnologice, profesorii sînt nevoiți să fixeze de la început parametrii experienței, pentru a fi siguri că se obține rezultatul așteptat. În noua tehnologie didactică, profesorul poate realiza mai întîi experiența după modelul clasic, simulînd-o apoi cu calculatorul — prin varierea parametrilor în limitele fenomenului studiat — pînă la înțelegerea și asimilarea sa deplină. Varietatea modelelor din realitatea practică, încă neidealizată, permite concretizarea abstractului, evidențierea mai rapidă a notelor comune esenței fenomenului sau procesului studiat. Conceptualizarea lor pe baza prelucrării și abstractizării în: definiții, teoreme, legi, proprietăți, reguli, metode și procedee de lucru, devine acum mai eficientă.

3°. Se știe că unele experiențe sînt dificile, uneori chiar imposibil de realizat în laboratoarele clasice. Aceasta se întîmplă fie datorită grafismului extrem de complicat, fie din cauza unor distanțe inaccesibile, a unor mișcări instantanee, sau a unor probleme privind protecția muncii și a mediului înconjurător.



Calculatorul poate fi folosit cu mare eficiență în aceste situații. Spre exemplu, există programe care simulează mișcarea corpurilor în spațiu — pe baza legii atracției universale, programe pentru studiul legilor dinamicii, al legilor lui Kepler, al legilor frecării, al mișcării browniene, al mișcării electronilor etc. În astfel de situații, folosirea grafismului permite vizualizarea fenomenelor fizice și observarea acestora în mișcare. Folosirea calculatorului cu tehnica sa de animație este de mare efect pedagogic în studiul transformărilor geometrice și al aplicațiilor acestora, în studiul grupului deplasărilor, în studiul locurilor geometrice și al secțiunilor conice etc. Astfel de simulări cu calculatorul sînt foarte productive în predarea geografiei, biologiei, tehnologiilor, a economiei, sau în studiile cu caracter statistic. De asemenea, experiența arată că prin intermediul programului didactic activitatea elevului este mai bine dirijată spre acțiune, explorare și descoperire personală.

4°. Calculatorul conferă tehnologiilor didactice un caracter interactiv. Prin intermediul programului didactic, profesorul „realizează“ un dialog permanent între mașină și cel care învață. Pentru ca acest dialog să fie cît mai plăcut, programele didactice trebuie să fie atractive, cu pauze, glume, ton și culoare, menite să-i creeze elevului o „microlume“ a sa în care să dorească să învețe știința, tot astfel cum învață să vorbească. Atît pentru profesor, cît mai ales pentru elev, limbajul de programare trebuie să fie cît mai simplu. Evident, limbajele convenționale fiind destul de sărace, este necesar ca profesorul să-i obișnuiască pe elevi să-și redacteze răspunsurile și în limbajul natural.

5°. Folosirea calculatorului solicită un studiu aprofundat al temei și spirit de cercetare din partea elevului. Necesitatea realizării schemei logice și a programului de rezolvare a problemei studiate solicită cunoașterea perfectă a teoriei. Conceptul de învățare și de aplicare capătă în acest context o nouă calitate — cu accent pe activitatea formativă, participativă și anticipativă a elevului. Folosirea calculatorului înlesnește o oarecare „dilatare“ a sferei de cuprindere a fenomenului studiat și, în același timp, o „comprimare“ a timpului folosit pentru studiu. În activitatea cercurilor școlare pe obiecte sau interdisciplinar, acolo unde este necesar și posibil, elevii sînt îndrumați să-și alcătuiască singuri schemele logice și programele corespunzătoare pentru folosirea calculatorului.

6°. Calculatorul acordă dreptul la eroare celui ce învață, punîndu-i la dispoziție comentariile și explicațiile necesare continuării lecției. Dialogul dintre mașină și cel care învață trebuie să fie cît mai apropiat de dialogul profesor-elev, deși nu se va putea atinge total această performanță.

7°. Însușindu-și în fața consolei o serie de cunoștințe, elevul se formează pe linia autoeducației permanente, învață să-și aprecieze calitatea și durata pregătirii, învață să învețe.

8°. Informatica și calculatorul facilitează spiritul algoritmic, programatic, operațional și organizatoric, rigoarea în raționament și exprimarea precisă, caracterul intensiv al muncii și spiritul aplicativ. Folosirea calculatorului poate optimiza învățarea rapidă a unor meserii în școli profesionale sau la locul de muncă, permite pregătirea pentru diverse profesii a celor handicapați etc. Calculatorul este astăzi nelipsit în munca de cercetare și conducere, în informarea documentară, în evidență..., în toate domeniile permeabile algoritimizării.

9°. Calculatorul, ca mijloc de învățămînt, permite o nouă formă de prezentare a lecțiilor, ilustrarea lor practică și integrarea cu cercetarea și producția. Calculatorul nu poate însă înlocui profesorul, după cum cartea n-a putut



face să dispară lecția la clasă și nici nu se poate substitui celorlalte mijloace de învățămînt. El constituie un mijloc de învățămînt în cadrul laboratoarelor școlare și trebuie folosit acolo (și numai acolo) unde poate contribui la optimizarea procesului de învățămînt, a spiritului aplicativ, de cercetare și producție.

10°. Există în lume o atmosferă propice înzestrării învățămîntului cu calculatoare, ceea ce este de bun augur. Există însă și unele voci tehnocrate care vor să desființeze școala, afirmînd că pentru fiecare familie „un calculator și un set de programe ar fi suficiente pentru un învățămînt autonom la domiciliu”. Există de asemenea și voci care dau alarma așa-zisei „înstrăinări” a individului, ca urmare a unui învățămînt automatizat. Noi spunem însă că educația nu se poate reduce la informatică și calculator, iar un învățămînt pe de-a-ntregul automatizat nu poate exista. Practica arată că, pe măsura dezvoltării tehnicii și a tehnologiei învățămîntului, sarcinile școlii și ale profesorului cresc, ele purtînd în vreme solia revoluționară a noii calități.

Este astăzi evident că apariția informaticii și calculatorului a sporit cu mult puterea științei și a tehnologiei, eficiența muncii științifice, posibilitatea informării și aplicării rapide în practică a conținutului informațional. INFORMATIA a devenit un factor important al progresului contemporan. Pregătirea organizată a tinerei generații în domeniul informaticii își va aduce contribuția la dezvoltarea componentei tehnice a învățămîntului și educației și la umanizarea continuă a acestei componente prin procesul muncii creatoare și al educației permanente.

### 15.3. Tipuri de laboratoare pentru procesul de învățămînt

Organizarea spațiului școlar pentru învățămîntul asistat de calculator nu este încă standardizată. Amintim, cu titlu informativ, cîteva variante:

1°. Clasele așa-zis automatizate sînt spații școlare amenajate pentru predarea-învățarea lecției în exclusivitate cu calculatorul. O clasă automatizată poate dispune de un calculator avînd circa 30 terminale, care recepționează lecția pe ecranul individual de la pupitrul fiecărui elev. Lecția ce se predă este stocată în memoria calculatorului. Procesul de predare se realizează printr-o succesiune de imagini, numite imagini-ecran, derularea lor fiind dirijată de către profesor — de la pupitrul instalației.

2°. Laboratoarele școlare actuale — pe discipline de învățămînt, sau chiar clasele obișnuite. În aceste spații se poate folosi calculatorul în procesul de predare a unor secvențe din lecția clasică, se pot propune exerciții de fixare și consolidare a lecției predate, se pot verifica lecții etc. Desigur, se poate realiza și învățarea individuală, fiecare elev avînd pe masa de lucru trusa informatică necesară. Tot aici se poate realiza învățarea în colectiv, pe grupe cu doi-trei elevi la o masă de lucru.

3°. Există și alte tipuri de încăperi, total diferite de actualele săli de clasă sau de laboratoarele școlare. În aceste „spații didactice” se realizează așa-zisa „autoinstruire automatizată”, sau „instruirea autonomă” — deci învățarea fără profesor.

Lecția este stocată în memoria calculatorului și se afișează pe ecran în ritmul pe care și-l reglează elevul. Cel care învață poate întrerupe această acti-



vităte și s-o reia mai tirziu — pe baza „recunoașterii“ sale de către mașină. Fiecare item al unei astfel de lecții poate fi însoțit de întrebări sau exerciții de control. În cazul răspunsurilor incorecte, cel care învață poate beneficia de funcția de ajutor a calculatorului. În acest mod, sînt puse la dispoziția utilizatorului (sub formă de imagini-ecran) comentariile și explicațiile necesare pentru formularea corectă a răspunsurilor. Astfel, orice eroare din procesul învățării este detectată și remediată pe bază de dialog. Calculatorul poate memora performanțele celui ce învață, timpul afectat învățării, răspunsurile eronate etc. Toate aceste elemente sînt necesare atît celui ce învață, cît și specialiștilor care se ocupă de optimizarea programelor didactice.

4°. Proiect facultativ pentru organizarea unei săli de laborator sau a unui cabinet pentru folosirea calculatorului în procesul de învățămînt.

În cele ce urmează vor fi prezentate proiecte pentru:

- sala de laborator și zestrea sa informatică;
- biblioteca anexă și tipologia programelor sale.

### 15.3.1. Sala de laborator (și eventualele sale anexe).

Laboratorul (sau cabinetul) de informatică poate fi organizat, sub o formă mai restrînsă, într-o sală de clasă obișnuită. Iluminarea sălii este dată de lumina zilei sau de o lumină artificială care bate spre alb-albăstrui. Pereții sălii și tîmplăria trebuie să aibă o culoare deschisă. Ferestrele să fie prevăzute cu draperii pentru a se regla vizibilitatea în timpul muncii cu calculatorul.

În laborator, pe peretele din față este așezată tabla de scris. Lîngă aceasta se află o tablă magnetică. La mijlocul tablei, deasupra acesteia, se află rulat ecranul de proiecție. Spre colțul din stînga și din dreapta acestui perete, la circa 2 metri înălțime și puțin spre interiorul sălii se află suspendate două monitoare TV color. Pe peretele din spate sînt așezate două corpuri de bibliotecă pentru tehnica de calcul și informatică, unde sînt păstrate programele informatice educaționale necesare asistării cu calculatorul a procesului de învățămînt. În sală se află 18 mese, echipate cu truse informatice, la care vor lucra 36 de elevi și o masă pentru profesor (figura 15.2).

Dacă se poate amenaja, în plus, o anexă a sălii de laborator în care să se poată pregăti metodic lecția informatizată, să se organizeze o microbibliotecă de specialitate sau o sală de studiu, aceasta ar contribui, desigur, la o mai bună funcționalitate a laboratorului.

De precizat că, în lipsa acestui spațiu cu rol de anexă a laboratorului, s-ar putea amenaja un **punct informatic** în spațiul destinat bibliotecii școlare. Acest „punct“ poate fi înzestrat cu o trusă informatică și cu programe, care să permită organizarea pregătirii individuale a elevilor care n-au fost prezenți la lecția informatizată. Evident această sarcină ar putea intra în competența și atribuțiunile de serviciu ale bibliotecarului. Este de menționat și faptul că din lipsă de spațiu laboratorul se poate amenaja și într-un laborator de fizică,..., cadrul cel mai adecvat fiind însă laboratorul de matematică.

Aparatura necesară sălii de laborator include:

1. Un tablou electric care asigură:

- 1°. alimentarea **postului de lucru** al profesorului;
- 2°. alimentarea **monitoarelor TV-color**, suspendate;
- 3°. alimentarea **celor 18 posturi de lucru** pentru elevi;
- 4°. **comanda ecranului de proiecție**;



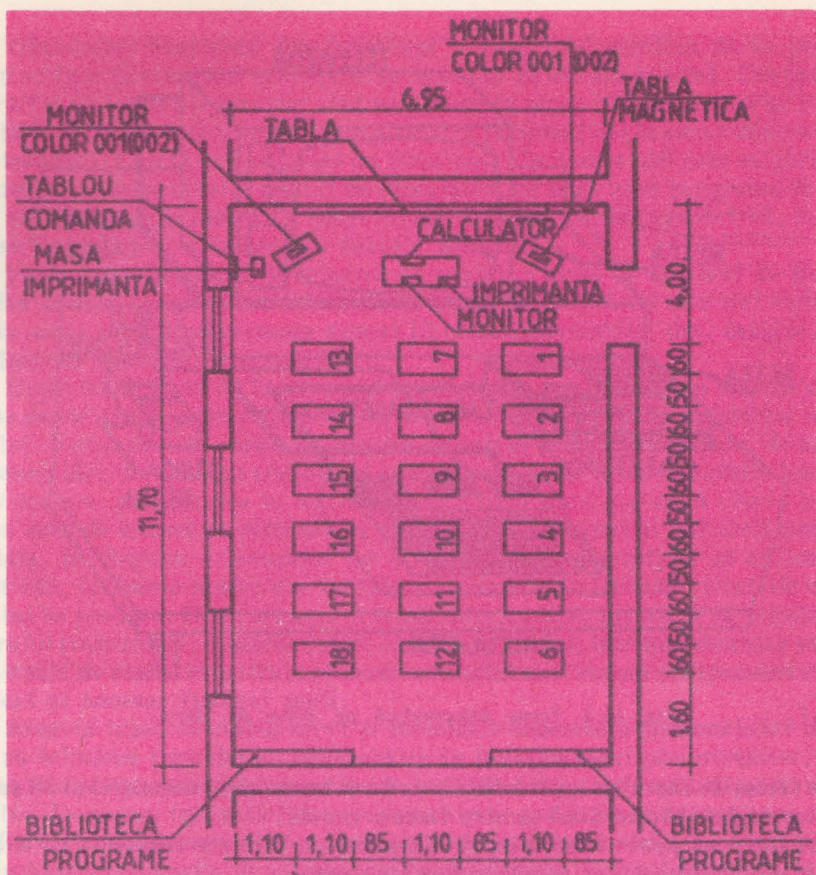


Fig. 15.2. Schița unei săli de laborator

- 5°. comanda draperiilor.
2. Mijloace de învățămînt optice ca:

- 1°. aparat de proiecție;
- 2°. diaproiector;
- 3°. retroproiector;
- 4°. aparat de proiecție cinematografică.

3. Echipament electronic pentru postul de lucru al profesorului și pentru cele 18 posturi de lucru ale celor 36 elevi ai clasei. Ca poziție în laborator, postul de lucru al profesorului se află în față, puțin spre stînga (spre fereastră) și este compus dintr-un pupitru (o masă cu legăturile electrice și accesoriile necesare) și-un scaun. Acest post funcționează independent și are ca echipament:

- 1°. un calculator personal HC 85, eventual cu disc flexibil și imprimantă (la postul profesorului s-ar putea folosi și calculatorul profesional FELIX PC.);
- 2°. un monitor color (sau alb-negru);
- 3°. un casetofon;
- 4°. o sursă.

Am menționa aici că, în perspectivă, este necesară la postul profesorului o imprimantă alfa numerică grafică, o interfață de comunicație și un microrobot. Posturile de lucru pentru cei 36 elevi ai unei clase sînt asigurate de 18



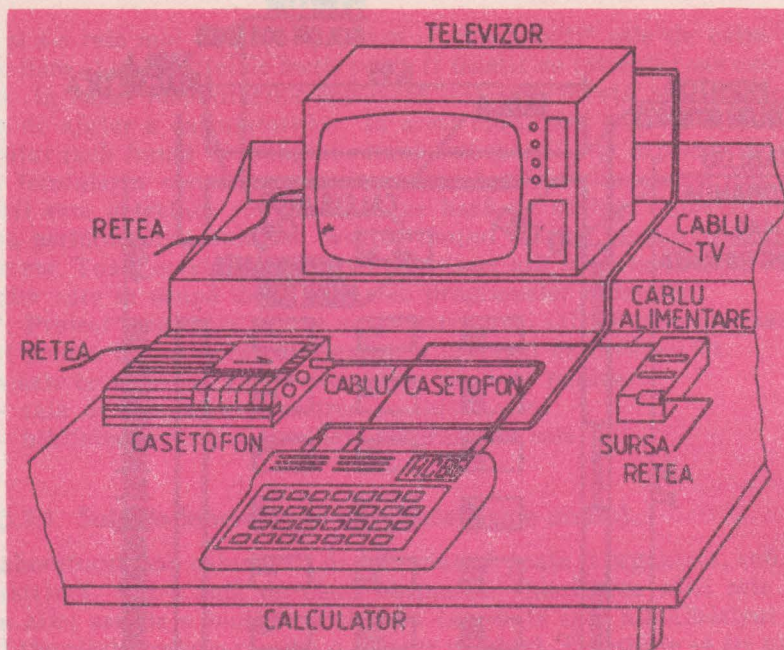


Fig. 15.3. Trusa informatică pe masa de lucru

pupitre (mese de lucru cu legăturile electrice și accesoriile necesare) și 36 scaune cu spătar reglabil (cîte două pentru fiecare masă).

Fiecare post poate funcționa și independent. El este înzestrat cu o trusă informatică alcătuită din următorul echipament:

- 1°. un calculator personal HC-85 (sau de alt tip);
- 2°. un televizor sport (în perspectivă un monitor color, alb-negru);
- 3°. un casetofon;
- 4°. sursa.

Evident monitorul de la postul de lucru al profesorului poate fi conectat cu oricare din posturile de lucru ale elevilor.

*Observație.* Avînd în vedere posibilitățile și fiabilitatea tipurilor de microcalculatoare electronice personale sau profesionale românești, utilizabile astăzi în procesul de învățămînt preuniversitar, cît și costurile acestora, apreciem că HC 85 este microcalculatorul performant și accesibil economic. Liceul Industrial „Dimitrie Cantemir” din București îl folosește în trusa informatică a postului de lucru (a se vedea figura 15.3).

Un post de lucru se pune în funcțiune urmărind figura 15.3. și indicațiile enumerate mai jos, astfel:

1. Se introduc în priză ștecherile: televizor, casetofon, sursă (conectorul de alimentare nefiind introdus în calculator);
2. Se alimentează calculatorul;
3. Se pornește televizorul;
4. Se fac legăturile: calculator-TV., calculator-casetofon.



**Observație:** Scoaterea din funcțiune a trusei informatice de la postul de lucru se face parcurgînd operațiunile de mai sus în ordine inversă. Desigur, dacă posturile de lucru ale elevilor sînt conectate în rețea locală, atunci se poate renunța la echipamentele pentru înregistrarea informației (în cazul descris de noi, casetofonul). Există săli de clasă cu terminale. În aceste săli este asigurată desfășurarea învățămîntului asistat de calculator, pe baza unor programe de conducere și îndrumare a instruirii individuale. Gîndim ca, în perspectivă, laboratorul să îndeplinească și această funcție didactică.

Facem, de asemenea, precizarea că pentru o clasă de 36 elevi se poate organiza munca în laborator folosindu-se numai 12 pupitre a cîte trei elevi fiecare. În acest mod sînt necesare numai 12 truse informatice, dar, desigur, cu un randament mai puțin intensiv.

Experiența arată că în procesul predării după metodologia clasică a lecțiilor se poate apela în mod frecvent la folosirea calculatorului ca mijloc de învățămînt. Oriunde este vorba de tehnică de calcul și de reprezentare, de simulări sau modelări, de grafism complicat sau de experiențe greu accesibile laboratoarelor școlare, prezentarea în lecția clasică a unor secvențe sau a cîteva imagini-ecran este de natură să conducă la optimizarea însușirii și aplicării noțiunilor predate, la lărgirea orizontului lor științific, interdisciplinar și aplicativ. În acest caz, în foarte multe din lecțiile noastre sînt solicitate virtuțile calculatorului în predare. Deci, o singură trusă informatică, instalată la pupitrul profesorului (calculator, monitor color, casetofon, sursă), într-un laborator sau chiar într-o sală de clasă, constituie în cazul semnalat mai sus un echipament eficient, modest și necesar fiecărei școli.

Această trusă ar putea servi și reciclării concrete a personalului didactic pentru folosirea tehnologiei informatice în învățămîntul preuniversitar, pentru alcătuirea programelor informatice de învățămînt în vederea utilizării calculatorului. Două sau trei astfel de echipamente ar putea constitui zestrea minimă imediată, necesară unei școli pentru activitatea informatică cu clasele a IX-a și a X-a, pentru cercurile de matematică sau interdisciplinare, pentru activitatea tehnico-productivă și pentru munca de cercetare a cadrelor și elevilor.

### 15.3.2. Biblioteca anexă

Ca și în cadrul celorlalte discipline școlare, literatura din domeniul informaticii și tehnicii de calcul intră în evidența și programul bibliotecii școlare. De aici nu pot lipsi, desigur, o serie de lucrări din domeniul logicii și al algebrei sau programării, al electrotehnicii, electronicii și automaticii. Pe noțiunile acestor discipline școlare se sprijină o înțelegere mai corectă atât a componentei mecanice și electronice („hardware”), cît și a componentei logice („software”) a calculatorului.

Biblioteca anexă laboratorului este destinată evidenței, conservării și utilizării programelor informatice educaționale (numite uneori și „didactice”), depozitării casetelor cu programe utilitare-organizate pe diverse discipline școlare sau pe capitole ale acestora, îndrumătoarelor pentru folosirea tehnicii de calcul din laborator, fișelor și scenariilor de lecții în care intervine calculatorul. Ea va trebui să aibă un fișier de programe, catalogate și contorizate, evidența lor făcînd obiectul unui program la care se poate apela ori de cîte ori se dorește o anumită rutină. Casetele cu programe vor fi ținute într-un dulap special,



casetă-catalog conținând titlurile tuturor programelor existente: număr de casetă, cifra de control la care se găsește programul pe această casetă, etc.

Casetă cu programul „catalog” poate fi apelată la nevoie, cerându-se titlul programului dorit. Răspunsul la apel oferă numărul casetei și locul unde se află programul pe bandă.

Evidența programelor didactice se ține pe discipline de învățămînt.

Oricare ar fi modalitățile de lucru privind organizarea și evidența programelor destinate procesului de învățămînt, vom distinge net două tipuri de programe:

1. Programe pentru profesor (privind procesul de cercetare și de predare);
2. Programe pentru elevi.

În cadrul programelor pentru elevi se pot distinge;

- a) programe monodisciplinare (pe obiecte de învățămînt);
- b) programe pluri sau interdisciplinare;
- c) programe pentru munca în producție;
- d) programe pentru activitatea de cercetare-proiectare etc.;
- e) programe pentru jocuri educative;
- f) programe-test pentru evaluări privind capacitatea psihomotorie a celor ce învață.

S-ar putea realiza și o altă clasificare a programelor informatice educaționale, de exemplu:

a) programe folosite de profesor în procesul de predare, acestea avînd ca suport psihopedagogic interacțiunea dintre imagine și concept în procesul de gîndire;

b) programe utilizate pentru învățămîntul autonom-programe de autoinstruire automatizată atît pentru elevi cît și pentru profesori.

În cadrul programelor didactice ale unei discipline de învățămînt este posibilă și o clasificare pe capitole ale respectivei discipline.

Tot în cadrul monodisciplinar este posibilă și o clasificare pe tipuri de programe, în concordanță cu metodologiile procesului de învățămînt. În acest sens, vom propune, în continuare, o variantă de clasificare a programelor informatice educaționale:

1. Programe (sau subprograme) care intervin în anumite momente ale lecției clasice de comunicare a cunoștințelor, de consolidare sau de verificare și control. În cadrul acestora se pot realiza exemplificări care folosesc modelarea, ori simularea unor procese sau fenomene expuse în lecție. Procedeu folosit, prin intercalarea în lecție a unor astfel de programe scurte, poate aduce o ameliorare a procesului de predare-învățare, dezvoltîndu-i eficiența didactică prin creșterea productivității activității cadrelor didactice și a elevilor.

2. Programe (sau subprograme) interactive de antrenament. Acestea se folosesc pentru consolidarea priceperilor și exersarea deprinderilor formate în cadrul unor lecții din domenii ale disciplinelor teoretice, tehnologice sau practice. În astfel de programe se folosește mult procedeul variațiunii pe aceeași temă și funcția de ajutor a calculatorului. Lecțiile acestui gen de programe sînt, oarecum, similare învățămîntului programat. Astfel de lecții oferă elevilor un surplus de experiență în studiu, pe baza muncii independente.

3. Programe pentru învățămîntul dirijat cu ajutorul calculatorului. În cadrul acestora se pot folosi: descoperirea dirijată, rezolvarea dirijată a problemelor, consultația programată, testarea și evaluarea cunoștințelor celor ce învață.



4. Programe pentru învățămîntul autonom. Și în cadrul acestor programe poate fi folosită descoperirea dirijată — pe baza unor întrebări de îndrumare (sau de control) și a sprijinului acordat de sistemul informatic. Programul este interactiv. Există în acest cadru și programe de autotestare cu evaluarea rezultatelor obținute. Programele pentru învățămîntul autonom sînt mai complexe, în sensul că ele prezintă și obiectivele învățării temei propuse, adeseori trecînd în revistă o serie de cunoștințe menite să formeze suportul noii teme. Pe parcursul noii teme sînt prevăzute teste pentru controlul însușirii noilor cunoștințe și corectarea eventualelor greșeli. În fine, programul se poate încheia printr-un autotest final, după care urmează uneori o serie de recomandări sau o temă propusă.

5. Programe de tip dialog (frecvent folosite în învățarea limbilor străine sau a limbajelor de programare).

6. Programe pentru jocuri. Acestea sînt specifice unei discipline școlare sau mai multor discipline. În acest caz se lucrează interactiv cu sistemul, folosindu-se mult simularea. Există jocuri simulatorii atractive pentru toate vîrstele școlare.

7. Programe pentru activitatea cercurilor de elevi (de exemplu programe de simulare în domeniul astronomic, mișcări kepleriene ale corpurilor cerești, lansări de nave cosmice, apariții de supernove sau găuri negre, etc.)

8. Programe pentru folosirea tehnicii informatice în munca de cercetare a cadrelor și elevilor. Programe—test pentru verificarea cunoștințelor la examene (concursuri).

9. Programe pentru evidența automată a muncii elevilor, pentru clasificarea acestora în cadrul colectivelor din care fac parte, pentru clasificări în cadrul concursurilor școlare, etc.

10. Programe de gestiune școlară și materială în domeniul unei discipline de învățămînt (sau a școlii însăși).

*Observație:* În predarea disciplinelor tehnice și a muncii productive utilizarea calculatorului electronic poate fi solicitată frecvent prin diferite tipuri de programe în care este necesară:

1°. Tehnică de calcul și de reprezentare, conducerea unor procese tehnologice și de producție;

2°. Modelarea unor procese sau fluxuri tehnologice complexe;

3°. Prezentarea succintă a unor operații de muncă mai dificile și repetarea acestei acțiuni pînă la însușirea lor corectă de către elev;

4°. Ilustrarea operațiilor tehnice și a locurilor de muncă caracterizate printr-o frecvență mărită de accidentare;

5°. Munca de cercetare a inginerilor, tehnicienilor și elevilor.

Pentru pregătirea modernă a forței de muncă sînt necesare, desigur, o serie de programe informatice pentru însușirea unor procese de producție în fabrică sau în atelierul-școală, la locul de muncă al elevului. Astfel, se pot realiza programe privind:

— conducerea sau supravegherea unor procese tehnologice în secții sau hale de producție (spre exemplu, conducerea unei mașini de imprimat sau urmărirea procesului de producție într-o secție de războaie de țesut — dacă ne referim la industria ușoară);

— centralizarea de date și realizarea operativă a unor evidențe pentru producția și aprovizionarea curentă sau pentru consumul de materii și materiale, în vederea realizării planului de producție;



- circulația produselor în decursul procesului de producție până la realizarea produsului final și depozitarea sau încărcarea și expedierea acestuia;
- sesizarea imediată a unor defecțiuni în procesul muncii, etc.

Este desigur evident că există un specific anume pentru fiecare disciplină de învățămînt și profil de meserie și de acest specific trebuie să se țină seama în alcătuirea programelor pentru calculator. Dar oricare ar fi sectorul de învățămînt — de la învățarea scrisului și pînă la reciclare sau cercetare, programele didactice pentru calculator sînt solicitate foarte mult acolo unde este vorba de:

- a) repetiție (respectîndu-se trecerea de la simplu la complex în prezentarea de variațiuni pe aceeași temă);
- b) simulare (oferindu-se prilejul modificării parametrilor subiacenți ai experiențelor — adeseori inaccesibile laboratoarelor școlare — prin aceasta încurajîndu-se raționamentul deductiv);
- c) modelare (încurajîndu-se descoperirea pe baza raționamentului de tip inductiv în studiul realității obiective);
- d) învățarea de tip programatic (cuantificîndu-se materia unei lecții pentru a facilita analiza, exercițiul de antrenament și decizia);
- e) conducerea unor procese de producție;
- f) jocuri (solicitînd spiritul programatic, adaptarea rapidă și decizia).

La finele acestei prezentări privind programele didactice pentru calculator și biblioteca anexă, precizăm că alcătuirea lor solicită munca în echipă a profesorilor și inginerilor, informaticienilor, psihologilor, pedagogilor și sociologilor. De calitatea colaborării acestor factori depinde și eficiența la clasă a programelor.

### 15.3.3. Propaganda vizuală pentru informatică și calculator în cadrul laboratorului

Evident, această propagandă depinde și de poziția laboratorului în clădirea școlii. Astfel, pe coridorul din fața sălii laboratorului se pot afișa o serie de planșe, gazete și fotomontaje, după cum urmează:

1. Planșe privind istoricul informaticii și calculatorului în date — pe plan mondial.
2. Planșe privind dezvoltarea informaticii și industriei calculatoarelor în România.
3. Planșe cu aspecte de la *Fabrica de Calculatoare și Întreprinderea pentru întreținerea și repararea calculatoarelor, alte instituții și centre de calcul.*
4. Planșe cu echipamente de calcul indigene.
5. Planșe didactice prezentînd blocurile schemei logice, instrucțiuni și comenzi ale limbajului „BASIC“.
6. Gazeta de perete a cercurilor de informatică.
7. Fotomontaj „Trofee elevilor în domeniul informaticii și calculatorului“.

În interiorul sălii de laborator se poate realiza un fotomontaj cu marile personalități din istoria informaticii și calculatorului, în lume și în țara noastră. De asemenea, în cadrul unui panou sînt afișate articole de Regulament privind munca, ordinea și disciplina, reguli de protecția și securitatea aparaturii și a muncii în laborator.



## 15.4. Rezolvarea unei probleme cu calculatorul.

### Etape; algoritmi și reprezentările lor; exemple de programe

#### 15.4.1. Etape necesare rezolvării unei probleme cu calculatorul

În rezolvarea unei probleme cu calculatorul este necesară parcurgerea următoarelor etape:

1°. Enunțarea problemei, evidențiind ce se dă și ce se cere.

2°. Analiza problemei cu scopul de a decide dacă este rezolvabilă cu calculatorul (în această etapă se studiază suficiența datelor, alegerea sau găsirea unei metode eficiente pentru obținerea soluției, alegerea unui limbaj de programare convenabil).

3°. Elaborarea unui algoritm de rezolvare a problemei, a cărui structură trebuie să fie în strinsă legătură cu modul de funcționare a calculatorului.

4°. Scrierea programului de rezolvare a problemei în limbajul de programare ales. Programul reprezintă descrierea algoritmului într-un limbaj de programare. El constă dintr-o succesiune de instrucțiuni care „indică” mașinii ce operații trebuie să realizeze, în ce ordine și asupra căror date. O comandă sau o indicație într-un program se numește instrucțiune. O instrucțiune se referă la o singură operație.

5°. Testarea și verificarea programului pentru înlăturarea unor eventuale greșeli.

#### 15.4.2. Noțiunea de algoritm.

Este știut faptul că matematica a acumulat pînă în prezent cel mai mare număr de algoritmi, constituiți ca metode de rezolvare a unor probleme. Informatica a preluat noțiunea de algoritm din matematică. Algoritmul este un concept intuitiv pentru rezolvarea unei probleme sau a unei clase de probleme. El constă dintr-o mulțime finită de operații aplicabile unor date inițiale, într-o ordine bine stabilită, producînd în timp finit rezultatele. Regulile de calcul alcătuiesc pașii algoritmului, iar succesiunea regulilor de calcul și a calculului se poate realiza grafic sub forma unei scheme logice (sau scheme bloc).

În principiu, partea cea mai delicată în rezolvarea unei probleme cu calculatorul constă în găsirea algoritmului de rezolvare și reprezentarea acestuia într-o schemă logică. Prezentăm, ca exemplu, algoritmul pentru aflarea valorii numerice într-un punct  $X_0$  a polinomului:

$$P(X) = a_0X^n + a_1X^{n-1} + a_2X^{n-2} + \dots + a_{n-2}X^2 + a_{n-1}X + a_n,$$
 valoare pe care o vom nota prin numărul  $S = P(X_0)$ . Pentru stabilirea algoritmului se scrie polinomul sub forma:

$$P(X) = (((\dots((a_0X + a_1)X + a_2)X + a_3 \dots)X + a_{n-2})X + a_{n-1})X + a_n$$

Această formă permite observarea succesiunii calculelor, pe care o reprezentăm printr-o suită de „pași”:



- Pasul 1.  $S = a_0$  (atribuim variabilei  $S$  valoarea  $a_0$ );  
 Pasul 2.  $i = 0$  (inițializăm variabila  $i$ );  
 Pasul 3.  $i = i + 1$  (variabila  $i$  primește valoarea  $i + 1$ );  
 Pasul 4. Dacă  $i > n$ , atunci oprim calculul, rezultatul fiind valoarea din  
 acel moment a lui  $S$ ; dacă  $i \leq n$ , atunci trecem la pasul următor;  
 Pasul 5.  $S = SX + a_i$ ; treci la pasul 3.

*Observații:*

1°. Variabila  $S$  ia consecutiv valorile:

$$a_0, (a_0X_0 + a_1), (a_0X_0 + a_1)X_0 + a_2, \dots, (((((a_0X_0 + a_1)X_0 + a_2)X_0 + a_3 \dots)X_0 + a_{n-2})X_0 + a_{n-1})X_0 + a_n = P(X_0).$$

2°. Variabila  $i$  ia consecutiv valorile  $0, 1, 2, 3, \dots, n$ , aceasta indicind a cîta paranteză urmează a fi calculată (în ordine:  $a_0, a_0X_0 + a_1$  etc). Ea este necesară pentru a indica momentul în care  $S = P(X_0)$  pentru  $i = n$ . Orice algoritm este caracterizat prin trei proprietăți:

- a). este general, în sensul că nu există problemă din clasa respectivă nere-solvabilă;
- b). este finit în spațiu (ca descriere) și în timp (ca execuție);
- c). este realizabil, ceea ce înseamnă că operațiile ce le comportă sînt efec-tuabile cu mijloacele de calcul disponibile.

#### 15.4.3. Reprezentarea algoritmilor prin scheme logice și prin limbaj de tip pseudocod.

Schema logică este reprezentarea sub formă grafică a algoritmului, precum și a modului de lucru al acestuia. Există o strînsă legătură între schema logică și programul corespunzător, în sensul că programul asigură efectuarea tuturor operațiilor prevăzute în schema logică. Schema logică este alcătuită din instruc-țiuni, acestea fiind încadrate în niște simboluri grafice sub forma unor figuri geometrice, numite blocuri. Prin intermediul acesteia, algoritmii de lucru sînt abordați în ideea programării structurate. În funcție de tipul instrucțiunii, blo-curile au denumiri specifice, descrise pe scurt și reprezentate în tabelul din figura 15.4.

1°. Bloc terminal sau bloc delimitator al programului. Indică începutul, respectiv sfîrșitul schemei logice, conținînd cuvintele START, respectiv STOP.

2°. Bloc de intrare-ieșire. O instrucțiune conținută într-un astfel de bloc se referă la transferul datelor. El pune în evidență informațiile ce se introduc (aso-ciindu-se cu citește (sau „read")), respectiv informațiile ce se extrag (cînd se aso-ciază cu serie (sau „write“)).

3°. Bloc de calcul. În blocurile de acest tip se inserează instrucțiunile de calcul și de atribuire. Spre exemplu, se calculează valoarea unei expresii și se atribuie unei variabile. El evidențiază deci calculele ce se efectuează și varia-bila care preia rezultatul.

4°. Bloc de decizie. Este un bloc de test, cu răspunsul DA sau NU, fiind folosit la realizarea structurilor alternative și repetitive (și numai așa). Are o intrare și două sau trei ieșiri. Acest bloc conține o instrucțiune de salt. I se mai spune bloc de test.




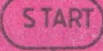
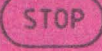
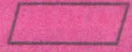

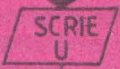

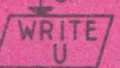

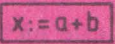
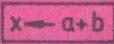

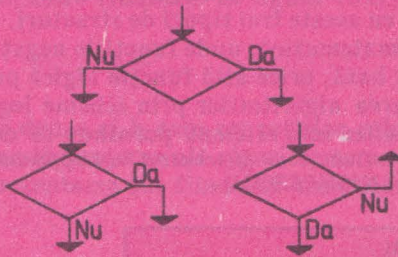


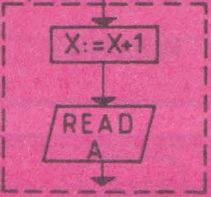

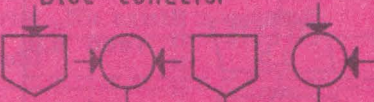



Nr. crt.	Simbol	Denumire și mod de utilizare
1		Bloc terminal  
2		Bloc de intrare/ieșire    
3		Bloc de calcul (atribuire)   Variabila X primește valoarea a+b
4		Bloc de decizie 
5		Bloc de secvență prescurtată (sau bloc de procedură)  
6		Bloc conector 
7		Săgeată  

Fig. 15.4. Blocuri logice



5°. Bloc de procedură (de prelucrare), sau bloc de secvență prescurtată. Este vorba tot de un bloc de calcul, semnificând o secvență din schema logică sau chiar o schemă ce se inserează. Se utilizează pentru evidențierea structurilor compuse, atunci când spațiul de scriere nu este suficient, respectiv pentru a marca apelurile de proceduri.

6°. Bloc conector (sau nod). Într-o schemă logică se folosesc și semnele grafice denumite „conector pentru aceeași pagină“, respectiv „conector pentru pagini diferite“. Conectorul pentru aceeași pagină este de tip cerc și leagă două puncte ale schemei logice evitând linia continuă — care ar încălca prea mult schema. Celălalt este conectorul pentru pagini diferite, permițând continuarea unei scheme logice pe o altă pagină.

7°. Săgeata. Indică modul de înlanțuire a simbolurilor ce compun schemele logice.

Facem aici precizarea că tehnica reprezentării algoritmilor prin scheme logice, are, față de alte metode, o mare facilitate vizuală. Acest fapt este foarte important în deosebi în faza de învățare. (O tehnică simplă este și cea a diagramelor).

Limbajul de tip pseudocod este un limbaj simplificat de reprezentare a algoritmilor. Pentru conceperea și reprezentarea algoritmilor se folosesc trei tipuri de structuri (secțiuni) de bază: liniare, alternative, repetitive (ciclice). Aceste considerate au la bază o teoremă care spune că orice algoritm se poate reprezenta cu aceste trei tipuri de structuri. Avem aici o situație similară cu cea din logica matematică, unde funcțiile negație, conjuncție și disjuncție sînt considerate de bază, orice altă funcție logică putîndu-se exprima prin acestea trei. Reprezentarea algoritmilor prin scheme logice și prin limbaj de tip pseudocod se poate realiza mai eficient pornindu-se de la exemple.\* (A se vedea „Ghid pentru predarea noțiunilor de informatică în învățămîntul preuniversitar“ M.E.I. București 1987, de STELIAN NICULESCU).

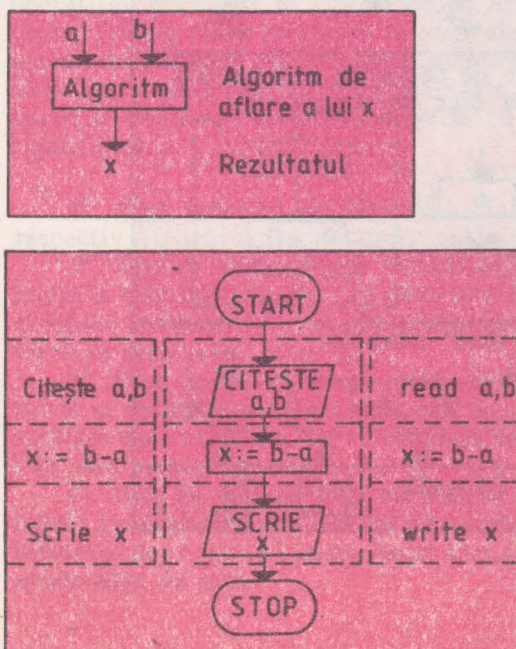


Fig. 15.5. a) Algoritm; b) Schema logică (Structură liniară); c) Reprezentarea în limbaj de tip pseudocod



## 1). Structuri liniare.

Fie, ca exemplu, rezolvarea ecuației:  $X+a=b$ ,  $a$  și  $b$  fiind numere naturale,  $a < b$ . Algoritmul de rezolvare a ecuației considerate se poate da sintetic așa cum se arată în figura 15.5.

Schema logică a algoritmului de rezolvare a problemei formulate este prezentată în figura 15.5 — a. În figura 15.5 — b se reproduce aceeași schemă logică, din figura 15.5 — a, dându-se alături de ea și reprezentările într-un limbaj de tip pseudocod (în limbile română și engleză).

Se remarcă ușor că algoritmul prezentat este:

- general, deoarece pentru orice  $a$  și  $b$  numere naturale, cu  $a < b$ , există  $x$ ;
- realizabil, deoarece operațiile pe care le presupune sînt realizabile cu orice calculator;
- finit în spațiu (ca descriere) și în timp (ca execuție).

Se observă că algoritmul în discuție se compune din trei structuri liniare elementare, efectuate în ordinea scrisă:

citește $a, b$	read $a, b$
$x := b - a$	$x := b - a$
serie $x$	write $x$

Se poate considera că algoritmul constă dintr-o singură structură liniară compusă din cele trei structuri elementare (indicate în figura 15.5 — c. prin liniile punctate).

## 2) Structuri alternative.

Dacă în problema anterioară se dau  $a$  și  $b$  care nu satisfac relația  $a < b$ , algoritmul nu dă  $x$  natural. De aceea se propune alt algoritm care să aibă în vedere și această eventualitate, reprezentarea lui fiind cea ilustrată în figura 15.6.

Scrierea în limba română și corespondența cu scrierea în engleză a reprezentării în pseudocod este:

citește $a, b$	read $a, b$
dacă $a < b$	if $a < b$
atunci	then
$x := b - a$	$x := b - a$
serie $x$	write $x$
altfel	else
serie „Nu există $x$ natural“	write „Nu există $x$ natural“
sfîrșit-structură	endif

Liniile punctate din figura 15.6 pun în evidență un nou tip de structură, cea alternativă, pe care am evidențiat-o (în aceeași figură) și printr-un limbaj de tip pseudocod.

În funcție de condiția  $a < b$  se efectuează:

$x := b - a$



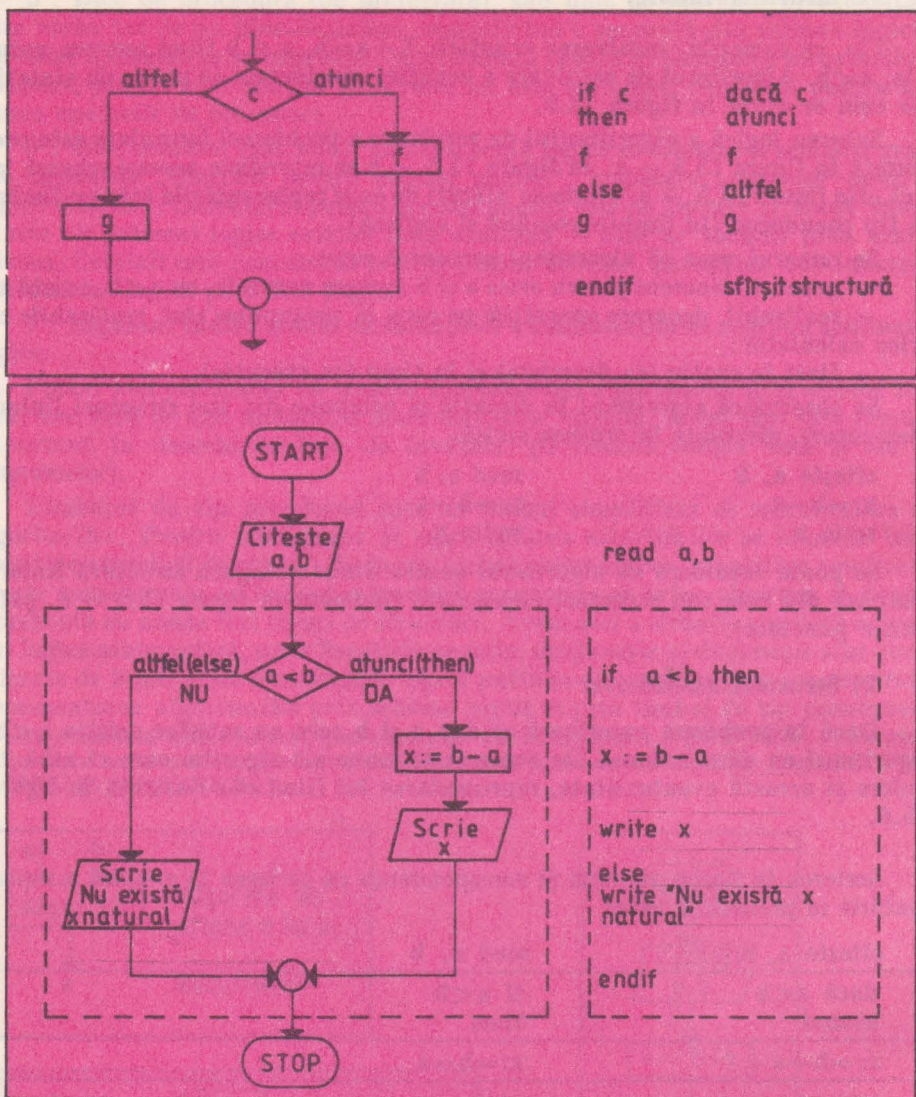


Fig. 15.6. a) Structură alternativă; b) Evidențierea structurii alternative

scrie x

atunci cînd condiția este îndeplinită; sau, se execută

scrie: „Nu există x natural“

în cazul cînd condiția nu este îndeplinită.

Se observă că o structură alternativă presupune o condiție c (în cazul considerat,  $a < b$ ) și două acțiuni f și g (în cazul considerat „f” fiind formată din structurile elementare

$x := b - a$  și

scrie x,



iar „g” din structura  
scrie: „Nu există x natural”).

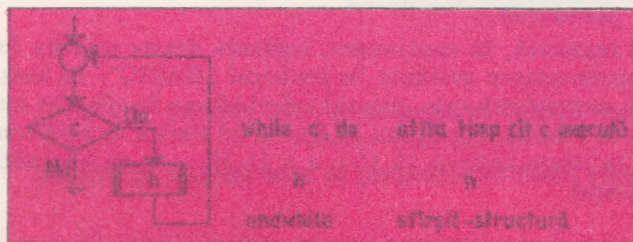
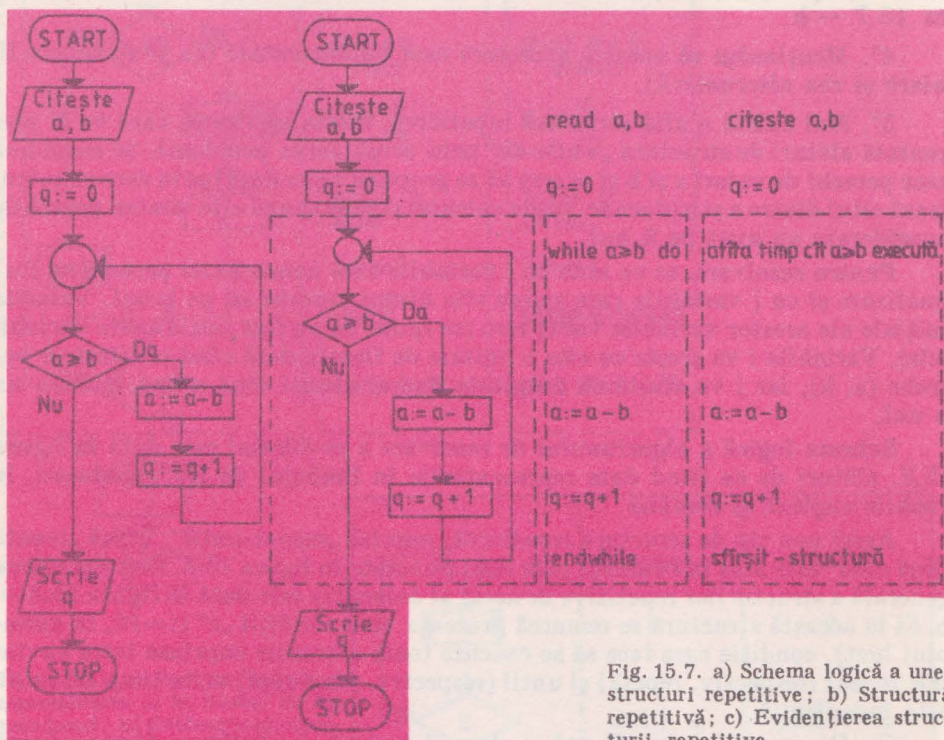
Reprezentarea structurii în maniera amintită este dată în figura 15.6 — a.

### 3). Structuri repetitive/

Ca și în cazul structurilor descrise mai sus, vom pleca de la un exemplu:

Fie a și b două numere naturale, cu  $b \neq 0$ . Să se determine, prin scăderi succesive, de câte ori se cuprinde exact b în a.

Soluție. Se scade mereu un b din valoarea lui a, atâta timp cât aceasta (mereu diminuată cu b) continuă să fie cel puțin egală cu b. Vom introduce o variabilă q pentru numărarea scăderilor succesive efectuate. Evident, pentru început, variabila q ia valoarea zero, căci formula de determinare a lui q este:  $q := q + 1$ . Vom spune astfel că variabila q are valoarea inițială zero. Rezolvarea sugerată în cuvinte este prezentată în figura 15.7. În figura 15.7 — a este reprodusă schema 15.7 însoțită de reprezentările într-un limbaj de tip pseudocod.





### Observații:

1°. Analizînd schema, se remarcă aici importanța presupunerii ca  $b \neq 0$ , ceea ce face ca numărul de scăderi succesive să fie finit (este asigurat faptul că algoritmul este finit). Am remarca, în plus, că se vede astfel de ce împărțirea prin zero nu este permisă.

2°. De asemenea, precizăm că valoarea lui  $a$  se pierde, ea fiind, în final, restul împărțirii întregi a lui  $a$  la  $b$ . Ca atare, se poate reconstitui valoarea lui  $a$  prin formula „ $a := b \cdot q + a$ ”, relație prin care se face proba împărțirii.

3°. Prin dreptunghiurile trasate cu liniuțe, din figura 15.7 —  $a$ , este pusă în evidență o structură repetitivă „while-do” (în engleză) sau „atîta timp cît — execută” (în limba română). O astfel de structură, după cum se remarcă, presupune o condiție  $c$  (care, în exemplul considerat, este „ $a < b$ ” și o condiție  $h$  (care, în exemplul considerat, constă din două acțiuni elementare, acestea fiind  $a := a - b$  și  $q := q + 1$ ). Reprezentarea structurilor de acest tip este redată în figura 15.7 —  $b$ .

4°. Menționăm că această structură este fundamentală (ca și structura liniară și cea alternativă).

5°. Mai există o altă structură repetitivă, foarte utilizată, care va fi prezentată alături de structura „while-do” prin următoarea problemă. Se consideră zece perechi de valori  $a$  și  $b$  și se cere să se propună algoritmul prin care se calculează cîtuș dintre  $a$  și  $b$  (dacă se poate calcula) și să se spună cîte dintre perechile considerate au avut pe  $b$  nul.

Pentru rezolvare, se va nota cu  $i$  variabila care numără cîte perechi au fost analizate și cu  $j$  variabila care spune cîte dintre perechi au pe  $b$  nul. Valorile inițiale ale acestor variabile vor fi zero (se spune că au fost inițializate variabilele). Variabila  $i$  va crește cu cîte o unitate de fiecare dată cînd se citește o pereche ( $a$ ,  $b$ ), iar  $j$  va crește cu o unitate numai atunci cînd  $a$  fost întîlnit un  $b$  nul.

Schema logică a algoritmului de rezolvare a problemei este dată în figura 15.8, alături de ea fiind date reprezentările în limbajul de tip pseudocod, în limbile engleză și română.

Acest nou tip de structură repetitivă, numită „repeat-until”, adică „repetă pînă cînd” este scos în evidență prin linii punctate în figura 15.8. Reprezentarea generală a structurilor repetitive de acest al doilea tip este dată în figura 15.8 —  $b$ . Și la această structură se remarcă prezența unei condiții „ $c$ ” ( $i = 10$ , în exemplul luat), condiție care face să se execute toate acțiunile cuprinse între cuvintele repeat (respectiv, repetă) și until (respectiv, pînă cînd) atîta timp cît ea nu este îndeplinită.

6°. Din reprezentarea făcută se observă în mod clar legătura dintre schema logică a algoritmului de rezolvare a unei probleme și reprezentarea algoritmului într-un limbaj de tip pseudocod.

În scop didactic va fi realizată, în continuare, legătura dintre schema logică a unui algoritm și reprezentarea acestuia în limbajul „BASIC”. În acest sens vor fi prezentate două exemple simple, primul referindu-se la *afișarea numerelor naturale impare de la 7 la 23*, (în figura 15.9), iar al doilea la *tipărirea numerelor naturale de la 1 la 10* (în figurile 15.10 —  $a$  și 15.10 —  $b$ , ca variante ale instrucțiunilor de testare).



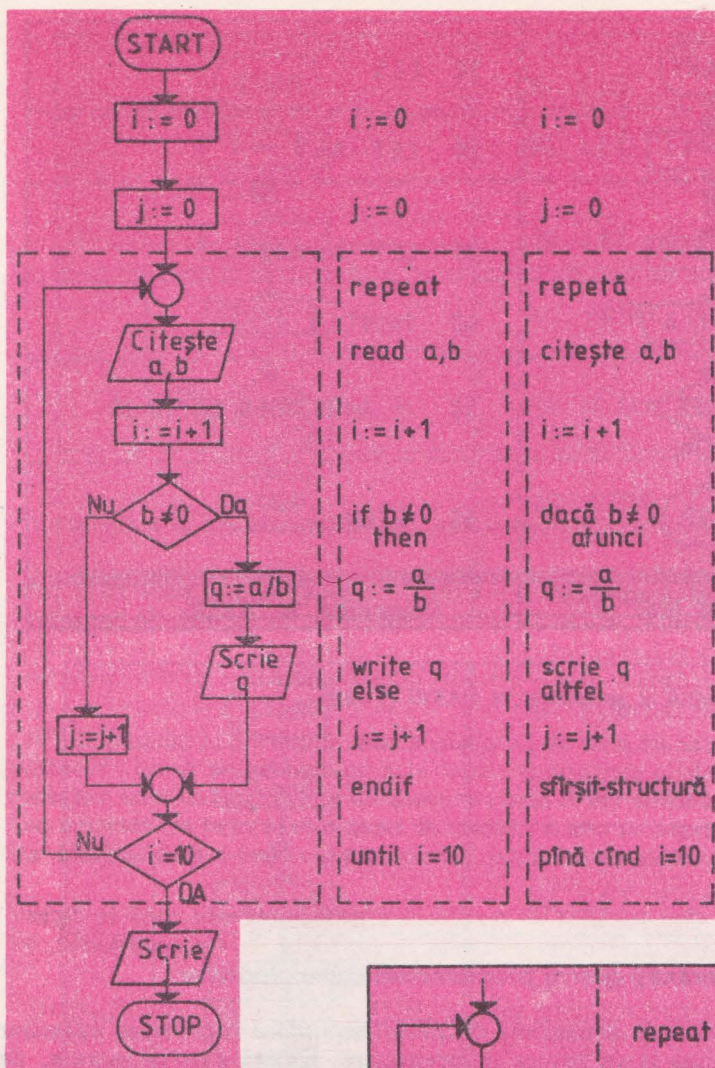
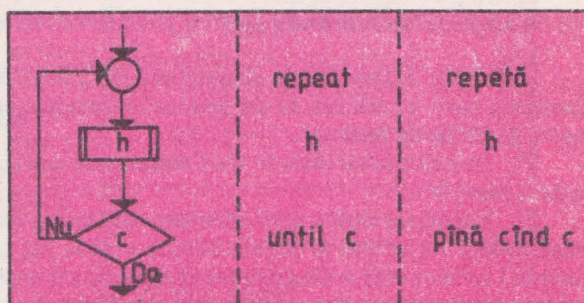


Fig. 15.8. a) Schema logică și reprezentarea ei în limbaj de tip pseudocod; b) Evidențierea structurii „repetă — până când”



Soluția 1. Schema bloc și programul BASIC corespunzător.

Soluția 2. Schema bloc și programul BASIC corespunzător.



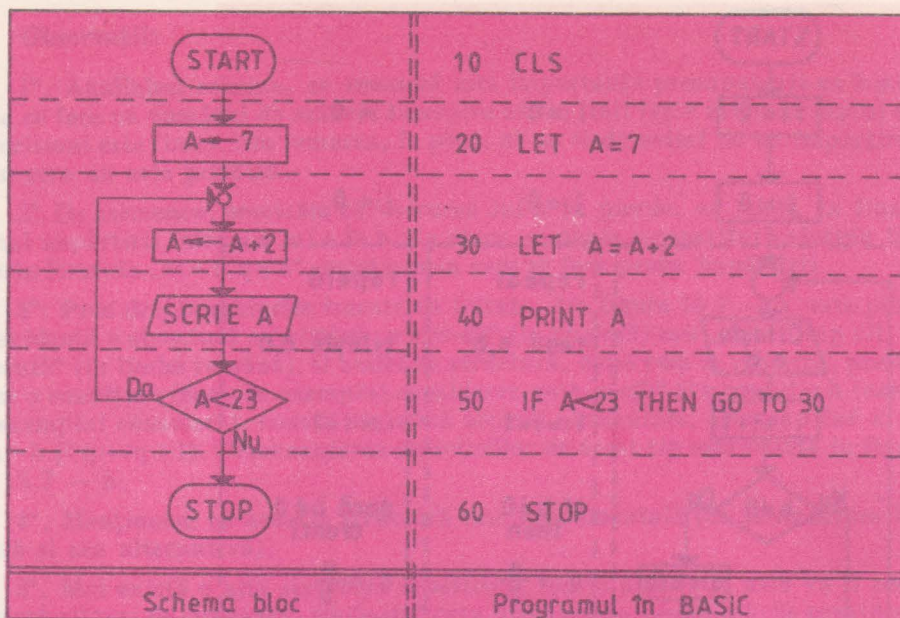


Fig. 15.9. Schema logică și programul în BASIC (7—23)

## 15.5. Scurtă sinteză privind elaborarea programelor

### 15.5.1. Cîteva noțiuni pentru programarea calculatoarelor

Noțiunea folosită sub denumirea de „date” semnifică totalitatea valorilor numerice care participă la un proces de prelucrare. Există date de intrare, de ieșire și date de lucru.

2°. În calculator datele sînt memorate codificat, în spații de memorie numite „locații”.

3°. Variabila este o cantitate care poate lua mai multe valori în desfășurarea unui proces de calcul și căreia îi este atașat un nume simbolic. Unei variabile simple i se asociază o singură locație. Numele unei variabile poate fi format din mai multe litere și cifre (cel mult șase). Primul caracter al numelui este o literă.

4°. În procesul de calcul intră și constantele.

5°. Cu ajutorul variabilelor și al constantelor se pot alcătui expresii aritmetice, folosind operații de adunare, scădere, înmulțire, împărțire, ridicarea la putere, extragerea rădăcinii..., precum și parantezele. De asemenea, se pot alcătui și expresii logice, acestea conținînd expresii aritmetice, constante și variabile logice, operatori logici.



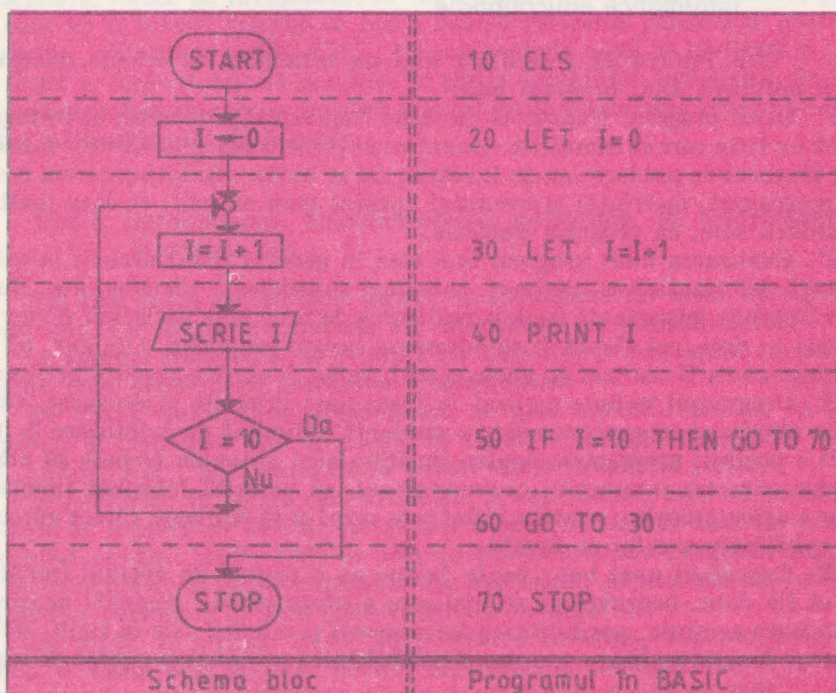
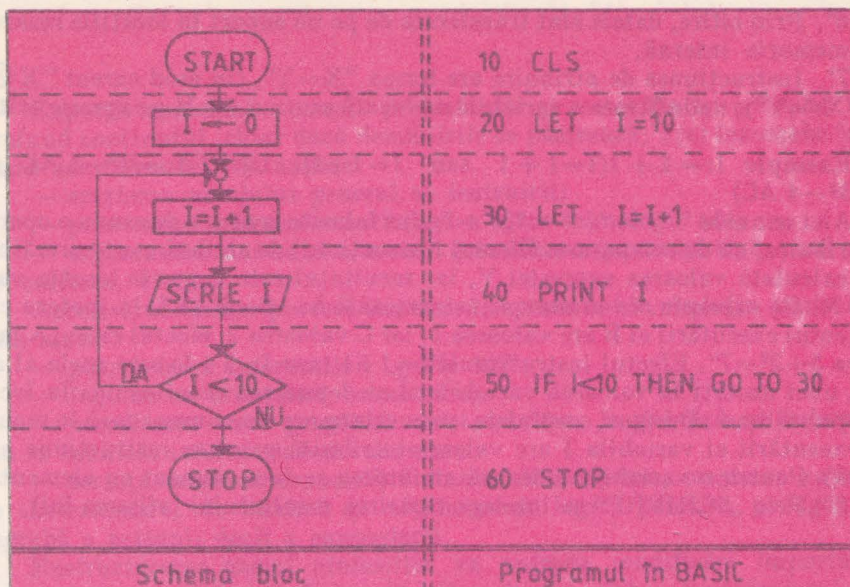


Fig. 15.10. a) Schema logică și programul în BASIC (1—10); b) Schema logică și programul în BASIC (altă variantă)



6°. Prin citire, datele sînt transferate de pe un suport în locațiile rezervate din memoria internă.

7°. Instrucțiunea de atribuire are forma " $X=Y$ ", (în mod curent " $X \otimes Y$ " sau " $X:=Y$ "), unde  $X$  este o variabilă indexată sau nu, iar  $Y$  o expresie aritmetică.

Exemple  $I:=I+1$  ( $I=I+1$  sau  $I \leftarrow I+1$ ),  $DELTA:=B^2-4AC$  (sau  $\Delta \leftarrow B^2-4AC$ )

Aici semnele " $:$ ", " $\leftarrow$ ", " $=$ " sînt folosite pentru a desemna o operație de atribuire, nu de comparare. Efectul instrucțiunii de atribuire este următorul: este calculată valoarea expresiei  $Y$ , iar rezultatul este depus în locația variabilei  $X$ . De exemplu, ca urmare a instrucțiunii  $A:=3 \times B+C^2$ , în ipoteza că în momentul executării ei  $B$  are valoarea 1, iar  $C$  valoarea 2, efectul este: „ $A$  ia valoarea  $3+2^2=7$ ". Efectul instrucțiunii  $I=I+1$  (sau  $I \leftarrow I+1$ ) este acela al adunării unei unități la valoarea variabilei  $I$  și depunerea rezultatului în locația corespunzătoare. Desigur, o astfel de instrucțiune are sens numai dacă în momentul executării ei variabila  $I$  are valoare (atribuită printr-o instrucțiune anterioară). Pentru o variabilă indexată atribuirea se face element cu element.

8°. Prin „SCRIERE” se înțelege tipărirea rezultatului (afișarea lui).

### 15.5.2. Citeva cerințe metodologice necesare realizării programelor informatice educaționale.

Vom încerca să prezentăm aici un minimum de cerințe obligatorii pentru definirea unui program bun.

1°. Orice program trebuie să fie bine fundamentat teoretic și pedagogic, să aibă un titlu care să semnifice semantic conținutul său și să fie ușor manevrabil în procesul de predare sau de învățare. Să se indice disciplina căreia îi aparține programul, instituția și eventual autorul (sau autorii) și data realizării și validării sale, ca program didactic.

2°. Realizarea unui program bun este în general rodul muncii în echipă: profesori, informaticieni, ingineri, psihologi și pedagogi, sociologi etc. Nu este de loc lipsit de importanță dacă la realizarea de programe participă și elevi sau studenți — deci, cei cărora li se adresează aceste programe. Desigur, un program bun poate fi realizat în unele cazuri, chiar de către o singură persoană.

3°. Programul trebuie realizat în ideea unei interfețe prietenoase, care să solicite atenția și concentrarea într-o ambianță optimistă de mobilizare la lucru. Pe cît e posibil, programele pentru învățămîntul autonom trebuie să aibă un puternic caracter interactiv, conversațional, să permită folosirea funcției de ajutor a calculatorului — atunci cînd este cazul și să realizeze corect aprecierea prin calificativ a celui ce învață.

4°. Cuprinsul unei teme poate începe cu o pagină de opțiuni (meniu) în care să fie date: instrucțiuni de utilizare a programului, comenzi necesare în exploatarea acestuia, posibilitatea întreruperii și reluării sale în timp, date de intrare și funcții realizate, precum și posibilitatea revenirii la pagina de instrucțiuni.

5°. Organizarea prezentării noțiunilor pe ecranul televizorului nu trebuie să-l obosească pe utilizator. Pentru aceasta se folosesc cu succes instrucțiunile BEEP, INK, PAPER, FLASH care dau programului o ușoară notă de muzicalitate și colorit agreabil.



## 15.6. Calculatoarele personale în învățămîntul din alte țări\*

Expertii apreciază că toate profesiunile și meseriile își schimbă în mod profund conținutul, chiar dacă denumirile se păstrează. Trei fenomene influențează aceste mutații și anume:

- creșterea nivelului general al instruirii;
- penetrarea microinformaticii și a altor tehnologii noi;
- organizarea mai bună a muncii (conducerea participativă, noul concept de calitate, etc.).

Accelerarea acestor procese ilustrează aspecte noi în evoluția profesiunilor, cum sînt:

- pătrunderea informaticii „peste tot“;
- înglobarea criteriilor economice;
- necesitatea unei pregătiri noi, simultan în tehnică și informatică, sau în tehnică, informatică și conducere;
- cerința creșterii capacității umane de adaptare, creșterea capacității de abstractizare, de analiză și raționament logic, printr-o pedagogie mai degrabă a acțiunii decît a cunoașterii.

Restructurarea gamei profesiilor va antrena modificări importante în sistemul de pregătire profesională. Deosebirile dintre învățămîntul tehnic și celelalte forme de învățămînt se vor atenua. În deceniul următor, ponderea în care va fi folosit timpul de lucru pentru ridicarea calificării personalului muncitor, va ajunge la 10—15%. Această metodă nouă de utilizare a timpului de lucru a început deja să se aplice în țările dezvoltate industrial, într-o serie de întreprinderi în care se implementează noile tehnologii bazate pe generalizarea utilizării calculatoarelor.

### 15.6.1. Introducerea calculatoarelor personale în învățămînt în diverse țări

Așa cum arăta un cunoscut expert, în acest final de secol educația reprezintă una din pirghiile esențiale ale dezvoltării pe plan mondial.

Ce sporuri însemnate, ce ritmuri ale dezvoltării economico-sociale se pot obține prin accelerarea proceselor educaționale?

De altfel, doi cunoscuți specialiști pe plan mondial arătau în 1984, că educația și instruirea într-un mediu de înaltă tehnologie reprezintă o investiție „intangibilă“, mai importantă uneori decît investiția fizică.

Într-adevăr, în multe țări socialiste și capitaliste, care au industrii dezvoltate de tehnică de calcul, dar și într-o serie de țări în curs de dezvoltare, începînd cu anii 1980—1985 au fost adoptate succesiv programe noi, ample, guvernamentale, la scara națională sau regională, pentru introducerea calculatoarelor personale în școli și instituții de învățămînt.

De pildă, în URSS, RPB, RPU, Marea Britanie, Japonia, Franța, Finlanda și India s-au adoptat decizii centralizate, iar într-o serie de alte țări s-au adoptat decizii diferențiate pe regiuni, ca în SUA, RFG și Canada, pentru a preciza numai o parte din țările care au opțiuni, preocupări, resurse și rezultate semnificative în acest domeniu.

\* A se vedea și anexa.



În cele ce urmează vom prezenta câteva date bazate pe publicații recente privind introducerea calculatoarelor personale în învățământ în câteva țări.

În URSS s-au adoptat măsuri care să asigure ca absolvenții de școli generale, profesionale, licee tehnice și învățământ superior să aibă o imagine clară a posibilităților oferite de tehnica de calcul și să poată programa într-unul din limbaje.

Preocupări privind instruirea asistată de calculator au existat încă din perioada 1950—1970, când s-au adoptat modificări ale planurilor de învățământ. În 1978, Ministerul Educației a aprobat un plan nou, cu o primă etapă până în 1982, referitor la introducerea sistemelor de predare asistate de calculator. În 1984, în cadrul „Direcțiilor de bază în reforma școlilor generale și profesionale” se precizează cerința „folosirii pe scară largă a calculatoarelor în procesul educațional”. În 1985, Biroul Politic al C.C. al P.C.U.S. și guvernul au aprobat prin decret, introducerea în toate liceele a unei materii noi „Bazele informaticii și ale tehnicii de calcul” pentru care s-a editat în tiraj de masă un manual, respectiv folosirea intensă a calculatoarelor în procesul de predare, precum și lansarea cursurilor de pregătire a cadrelor didactice.

A fost înființat Comitetul de Stat al URSS pentru tehnica de calcul și informatică și există un plan pentru introducerea în școli, în acest cincinal, a peste 500 de mii de calculatoare personale. Acest comitet dispune de un centru de informatică, în capitală, care acordă o gamă largă de servicii populației și întreprinderilor, având în vedere extinderea serviciilor în toate regiunile. În vacanțele școlare se organizează cursuri de pregătire pentru cadrele didactice și tabere de instruire pentru copii, iar la Palatul pionierilor din Moscova s-a înființat un cerc de informatică pentru elevi din clasele V și VI, care primesc diplome de programatori, după 3 ani de activitate. A fost adoptată o hotărîre a C.C. al PCUS și a Consiliului de Miniștri al URSS „Cu privire la asigurarea instruirii elevilor din școlile medii în utilizarea calculatoarelor și introducerea pe scară largă a tehnicii electronice de calcul în procesul de învățământ”. În vederea asigurării continuității pregătirii pe toate treptele de învățământ, ministerul unional al învățământului superior a stabilit trei niveluri pentru pregătirea de bază a specialiștilor neinformaticieni în utilizarea calculatorului (pentru informaticieni fiind prevăzute forme avansate de instruire):

- primul nivel implică pregătirea în proporții de masă a utilizatorilor, familiarizarea cu aplicațiile calculatoarelor și microprocesoarelor, programarea într-un limbaj. Studenții trebuie să rezolve pe durata școlarizării 10—15 programe (deci cite 1—2 pe semestru), fiind alocate 60—100 de ore de curs, din care peste 50% ore de lucru pe calculator;

- la nivelul al doilea se realizează o adîncire a specializării, fiind dezvoltate cunoștințele despre arhitectura microsistemelor, programe și algoritmi;

- la nivelul al treilea, specialistul asimilează metode avansate de programare și de optimizare, fiind în măsură să utilizeze eficient pachetele de programe aplicative existente sau să elaboreze programe noi.

Sînt prevăzute niveluri diferite ale pregătirii de bază, pentru diverse specialități ca de pildă: pentru agricultură și sivicultură — nivelul 1; pentru energetică, metalurgie, tehnologie chimică, tehnologia bunurilor de consum, transporturi, construcții — nivelul 2; pentru electronică, aparate electrice și automatică — nivelurile 2 și 3; pentru construcții de mașini și economie — toate cele trei niveluri.



În această etapă eforturile sînt concentrate pe următoarele obiective:

- asigurarea unui program flexibil de instruire, care să nu fie dependent de tipul calculatorului;
- reciclarea rapidă a profesorilor;
- dotarea unităților de învățămînt mai ales cu calculatoare personale și microcalculatoare.

*În R. P. Bulgaria*, Comitetul pentru Știință și Tehnologie, Academia de Științe și Ministerul Educației au inițiat primele acțiuni de realizare a calculatoarelor personale proprii Pravets (compatibile cu modelul Apple II) și de introducere a acestora în școli, în perioada 1979—1981. Astfel, în 1985 au fost introduse 6000 de calculatoare personale în școli, iar pînă în 1990 se vor introduce 40000 bucăți.

A fost înființat un Grup de cercetare în domeniul educației asistate de calculator, fiind experimentată predarea informaticii din clasa I-a, în cadrul cunoștințelor generale și din clasa a V-a ca materie distinctă, pe baza unor manuale elaborate în mod adecvat.

În cadrul programului național pe termen lung (pînă în 1990), începînd cu anul școlar 1986/1987, cursul de trei trimestre de informatică este obligatoriu în toate liceele, acestea fiind dotate inițial cu cîte 10 calculatoare personale. Sînt organizate totodată cursurile de specializare pentru profesori, care au durată variabilă: o săptămîntă, o lună, trei luni și un an.

Organizația de tineret din RPB a înființat în toată țara cercuri de informatică, iar televiziunea transmite lecții de inițiere în limbajul BASIC.

*În R. P. Ungaria*, Ministerul Culturii și Învățămîntului a luat decizia și a alocat în 1983 resursele necesare pentru introducerea calculatoarelor în învățămîntul mediu. Pînă la finele anului 1984, 3000 de calculatoare personale pentru uz școlar (modelul HT 1080 Z produs în Ungaria) au fost donate școlilor medii. Din anul 1985, școlile medii au început să fie dotate cu calculatoare tip Commodore 16. Numărul total al calculatoarelor aflate în școlile primare și secundare în 1985 era estimat la 7000.

Cele mai multe aplicații școlare erau, în ordine: în matematică, fizică, tehnologie, cercuri de studii, limbi străine, statistică, administrație, chimie, istorie, mecanică.

În perioada inițială, cele mai multe utilizări ale calculatoarelor aveau în RPU, un caracter demonstrativ și erau folosite mai ales în cadrul cercurilor de studii. O preocupare deosebită a profesorilor este aceea de a alege una din teoriile posibile, care interpretează învățarea ca o interacțiune continuă (dialog permanent) între elev și sistem, integrînd calculatorul în cadrul lecției la clasă.

Funcția calculatorului în tehnologia învățării trebuie să fie nu numai pentru exerciții recapitulative și practică, ori pentru jocuri și simulări, ci și pentru prelucrarea unor cunoștințe noi și pentru rezolvarea de probleme în cadrul acestora.

Există un program, prin decizie la nivelul guvernului, pentru introducerea sistemelor video, în școli, sub coordonarea Centrului național pentru tehnologie educațională. Se apreciază astfel, că se vor obține rezultate deosebite prin cuplarea celor două sisteme de instruire, asistat de calculator și bazat pe echipamente video.

## 15. C.P. ÎN ÎNVĂȚĂMÎNT



Și în alte țări socialiste ca RSC, RDG, RPP, RSF Jugosлавia și RP Chineză, se fac mari eforturi pentru accelerarea introducerii calculatoarelor personale și microcalculatoarelor în procesul de învățămînt și educație.

Datorită condițiilor istorice, al ordinii economice inechitabile existente și al avansului tehnologic al țărilor capitaliste „deszoltate” (subiecte cunoscute, dar care nu fac obiectul cărții), introducerea calculatoarelor în învățămînt a început mai de mult în acestea (iarăși, nu discutăm aici de coexistența flagrantă în mai multe țări „deszoltate” a calculatoarelor cu analfabetismul).

În SUA primele activități sistematice de introducere a calculatoarelor în procesele educaționale datează din anul 1966, cînd s-au alocat la nivel guvernamental fonduri de 150 milioane \$ pentru perioada pînă în 1971. Totuși, în SUA, datorită sistemului educațional foarte descentralizat, nu există un plan general de introducere a calculatoarelor în școli, în întreaga țară, astfel încît fiecare stat adoptă o strategie proprie (incluzînd donații, stimularea eforturilor proprii ale școlilor, eşalonarea plăților pe 3 ani, reducerea taxelor pentru producători-donatori, etc.).

Parcul calculatoarelor personale instalate în școlile din SUA a evoluat rapid:

- 33 mii în 1981;                      — 630 mii în 1984;                      — 1200 mii în 1987.
- 130 mii în 1982;                      — 700 mii în 1985;

Numai în 1983—1984, costul calculatoarelor introduse în școli a fost de 200 milioane \$.

Încă din 1985, 90% din școli aveau cel puțin un calculator personal, iar în 1987 în școlile secundare, la fiecare echipament reveneau 20 de elevi.

Peste 8% din școli au minim 15 calculatoare personale, iar 10% din școlile secundare au conectat calculatoarele la o rețea.

În mediul universitar există chiar cerința ca în viitor fiecare student să aibă acces la un microcalculator.

În medie, pentru activitatea cu calculatorul se alocă săptămînal 23 minute/elev în școlile elementare și 45 de minute/elev în școlile secundare.

În școlile elementare ponderea instruirii asistate de calculator din timpul de utilizare a calculatorului este de 50%, spre deosebire de școlile secundare la care devine dominantă ponderea elaborării de programe noi, față de asistarea instruirii. Producția de software educațional, inclusiv pentru școli, este asigurată de circa 750 de producători care dispun de structuri adecvate de desfășurare a produselor-program.

Din totalul programelor elaborate cele mai multe sînt în domeniile: matematic (18%), artistic (21%) și științific (11%).

În 98% din școli se predă limbajul BASIC, iar în 5% din școli și limbajele LOGO, PASCAL, FORTRAN, ponderea acestora fiind în creștere.

Cele mai multe calculatoare personale în școlile din SUA sînt de tip Apple (peste 50%), dar în școli și mai ales în universități, pătrund rapid modelele de tip IBM PC.

În Marea Britanie a fost adoptat în 1980 de către departamentul de resort, un program intitulat „Programul național de educație cu ajutorul micro-electronicii”. Începînd din 1980—1982, prin subvenții de la Departamentul pentru Comerț și Industrie (în cadrul programului „Microcalculatoarele în școli”), și prin eforturi proprii combinate cu un ajutor guvernamental (în proporție de 50%, pentru primele 2 microcalculatoare de fabricație engleză), au fost dotate:

— 7000 de școli secundare frecventate de elevi de 12—18 ani (în medie cu cîte 4 calculatoare personale);

— 27000 de școli primare pentru copii între 5 și 11 ani.



Fondurile guvernamentale alocate acestui domeniu depășesc 50 milioane lire și sînt împărțite astfel: 1/3 pentru acțiuni la scară națională și 2/3 pentru inițiative pe plan regional. Centrele regionale de pregătire a profesorilor oferă cursuri de 3 zile, 1 săptămînă și 3 luni, iar fiecare școală care s-a dotat cu un calculator personal cu ajutorul guvernului, trebuie să înscrie cel puțin 2 profesori la unul din cursurile respective.

Printre cele mai răspîndite modele sînt Acorn BBC și Sinclair ZX Spectrum. Fiecare școală a primit și patru seturi de programe, însoțite de manuale și două casete magnetice cu programe.

Elaborarea unui software adecvat, care să poată fi integrat în programa școlară, constituie o preocupare majoră în învățămîntul britanic. Astfel, grupul Dudley, compus inițial din 25 de profesori, a emis idei pe diverse tematici, iar specialiștii le-au transformat în programe care au fost testate și apoi publicate. De altfel, nu numai televiziunea prin emisiuni educative pe teme de informatică și microelectronică, dar și marile edituri acordă o atenție tot mai mare cărților despre calculatoare și programe, pentru care se estimează în viitor o pondere de circa 10% din totalul vânzărilor de cărți. Pentru copii între 6 și 14 ani sînt comercializate cca. 250 de programe cu un cost mediu de 10 lire/program.

Se acreditează tot mai mult ideea după care, pentru a constitui un factor de progres în procesul de învățămînt, calculatorul trebuie să fie însoțit de un software pe deplin corespunzător obiectivelor respective. Mai mult chiar, un software neadecvat poate să-i facă pe cei mai buni profesori să predea lecții slabe.

În alte țări dezvoltate industrial se fac de asemenea eforturi financiare și organizatorice considerabile pentru introducerea calculatoarelor în școli și instituții.

În Japonia exista în 1983/1984 următoarea distribuție a implementărilor de calculatoare personale în școli și în centre de educație:

- aproape 60% din școlile secundare (marea majoritate fiind cele de curs superior), cu peste 4 calculatoare în medie pe școală, 2% din școlile secundare avînd peste 20 de calculatoare pe școală;

- circa 0,6% din școlile elementare, în medie cu aproape două calculatoare fiecare;

- 0,3% din grădinițe, în medie cu un calculator;

- peste 10% din centrele pentru copii, în medie cu peste 6 calculatoare.

În Japonia, copiii vin în contact cu calculatoarele încă de la grădiniță, iar utilizarea calculatorului în educație face parte din strategia națională privind informatizarea societății. De altfel, a fost înființat un comitet la nivelul conducerii guvernului, cu sarcina de a elabora un plan de modificare radicală a sistemului educațional.

În Franța, în 1984, o mie de licee și o mie de institute și școli de învățămînt superior erau dotate cu calculatoare personale. Parcul calculatoarelor personale din școlile medii era în 1984 de circa 35000, la care se adăugau 120000 în 1985, astfel încît pînă la finele anului 1986, toate liceele, respectiv pînă în 1988, toate universitățile, urmau să fie dotate în medie cu 4—16 calculatoare.

Elaborarea și difuzarea programelor este coordonată în Franța de Centrul Național al Documentelor Pedagogice, care avizează printr-un consiliu de specialiști programele cele mai adecvate, le multiplică și le difuzează în școli.

Deși în 1985, a fost lansat în Franța programul „Informatica pentru toți”, acest program suferă după părerea ministerului educației naționale, de „inexis-



tența unei veritabile pedagogii a utilizării calculatoarelor". Cu ocazia lansării în 1986 a noilor proiecte de introducere a informaticii în domeniul învățămîntului, acesta arăta că procesul trebuie să răspundă la două cerințe principale:

- o cerință economică, deoarece învățămîntul național oferă o piață fabuloasă care poate stimula industria de calculatoare;

- o cerință pedagogică, deoarece copiii trebuie pregătiți pentru asimilarea noului salt tehnologic pe care îl aduce informatica.

Organizarea acțiunilor este bazată în Franța pe următoarele elemente:

- mutarea accentului de pe predarea programării, pe înțelegerea utilității informaticii, dezvoltînd două funcții pedagogice simple: simularea și „raportul nou cu știința”, care rezultă din utilizarea băncilor de date;

- inițierea din septembrie 1986, a unui concurs național de proiecte de elaborare de calculatoare pentru învățămînt și de concepere a caietului de sarcini, elaborare care a fost prioritatea bugetului învățămîntului pe 1987, beneficiind de o sumă de circa 100 mil. franci;

- alocarea în 1987 a circa 300 mil. de franci pentru implementarea în rețeaua învățămîntului a 20 000 de microcalculatoare.

Iată cum a evoluat în Franța numărul de calculatoare personale instalate în școli și numărul de profesori instruiți, respectiv fondurile alocate, pe etape:

- proiectul celor 58 de licee dotate cu câte un microcalculator, în 1970—1976;

- proiectul introducerii a 10 000 de microcalculatoare în școlile secundare, în 1979—1984. În 1982 erau instalate 3000 de microcalculatoare și erau specializați 19000 de profesori;

- proiectul introducerii a 100 000 de microcalculatoare în 1983—1988, în școlile secundare și elementare. În 1983 au fost instalate încă 12000 de microcalculatoare și specializați 11000 profesori (în cadrul fondurilor de 18 milioane \$), iar în 1984, 25000 de microcalculatoare introduse și 20000 de profesori specializați (45 milioane \$);

- proiectul introducerii altor 120000 de microcalculatoare în toate școlile secundare și elementare și specializarea a 100000 de profesori în anul 1985 (circa 200 milioane \$). Dintre aceste microcalculatoare 14000 erau compatibile cu tipul IBM-PC.

În 12000 de școli, la modelele IBM-PC sînt conectate în medie cîte 6 calculatoare personale.

Astfel, în Franța erau instalate în total pînă în 1986, 160000 de calculatoare personale și microcalculatoare, (populația școlară fiind de aproximativ 12 milioane) și erau specializați 150000 de profesori (circa 25% din totalul cadrelor didactice).

*În Finlanda*, guvernul a lansat acțiuni de studiere a noii tehnologii educaționale încă din 1979, comisia înființată recomandînd introducerea cursurilor de informatică la elevii între 9 și 12 ani și adoptarea unor măsuri pentru dezvoltarea tehnologiei respective. În 1985, Ministerul Educației a inițiat un proiect de cercetare la scară națională, intitulat „Calculatoarele și predarea”, pentru care numai în anul 1987 era alocat un fond de 8 milioane \$.

În școlile generale secundare din Finlanda există 3000 de calculatoare personale (circa 6—7 calculatoare/scoală), iar în școlile specializate circa 3500.

Începînd din 1987, există o varietate de cursuri de informatică, unele obligatorii, altele opționale, pentru elevii din școlile secundare și cu profil specializat.



Din totalul de 75000 profesori, peste 32000 au urmat cursul de 3 zile, 12000 pe cel de 3 săptămâni și 1500 au absolvit cursul de 4 luni.

De asemenea, au fost organizate în școlile primare peste 100 de cercuri de informatică.

*În India*, s-a pornit de la ideea că educația modernă incluzând informatica, cu alte cuvinte pregătirea corespunzătoare a forței de muncă, are un rol foarte important în dezvoltarea economică.

Pe de altă parte, deoarece în India marile fabrici și întreprinderi s-au automatizat și „informatizat” a apărut o cerere foarte mare de informaticieni cu pregătire medie și superioară.

Ca urmare, începând cu anul 1985 s-a declanșat un Program național de introducere a informaticii în învățământul liceal și universitar din India.

Concomitent a început producția de microprocesoare pe baza achiziționării tehnologiei din străinătate și apoi, India a început să producă și calculatoare personale compatibile IBM.

Pentru început experimentul educațional a fost declanșat în 250 de licee și în câteva școli generale. În anul 1986, experimentul s-a extins la 750 licee, vizând apoi o nouă extindere rapidă la 10000 de unități școlare. Astfel, în anul 1987 s-a aprobat un Program de organizare și dotare a învățământului profesional în domeniul informaticii și tehnicii de calcul.

S-au declanșat programe de cercetare a problemelor legate de introducerea calculatorului în învățământ, de stabilirea tehnologiei corespunzătoare și a folosirii adecvate a acesteia.

Una din concluziile desprinse din experiența de până acum a fost, aceea că pentru instruirea tinerei generații în domeniul informaticii, factorul cheie îl constituie profesorul. De aici a rezultat necesitatea obiectivă a:

- instruirii profesorilor existenți;
- formării corespunzătoare a noilor generații de profesori;
- elaborării unor materiale necesare instruirii periodice și continue a tuturor categoriilor de profesori.

Ca urmare s-au înființat pînă în anul 1987, în diferite universități, peste 42 de centre de instruire și formare a cadrelor necesare.

\*

**Experiența dobîndită pe plan mondial în domeniul instruirii școlare asistate de calculator permite să se afirme deja că există o afinitate naturală a copiilor pentru noua tehnologie.**

De altfel, din cele două manifestări internaționale de prestigiu pe tema utilizării calculatoarelor personale în educația tinerei generații, ambele organizate sub auspiciile UNESCO și găzduite de două țări socialiste, Ungaria, respectiv Bulgaria, în 1984 și 1985, Conferința și expoziția internațională din Bulgaria s-a reeditat după doi ani, în mai 1987, la Sofia, sub titlul sugestiv: „Copiii în era informaticii: oportunități pentru creativitate, inovație și noi activități”. La această manifestare științifică au fost prezenți peste 400 de specialiști din 44 de țări din întreaga lume. Cele peste 40 comunicări științifice au abordat următoarele trei subiecte mari:

- a) Către o nouă semnificație a alfabetizării;
- b) Informatica în școală și în afara școlii;
- c) Teoria educațională și sistemul cunoștințelor de bază.



S-au prezentat și discutat diferite forme și metode privind însușirea cunoștințelor de bază, formarea și consolidarea deprinderilor practice de utilizare a calculatorului, rolul acestuia în stimularea creativității tinerelor generații. În urma acestor dezbateri s-au desprins următoarele concluzii:

— ținând cont de faptul că în toate domeniile de activitate s-au introdus sau se vor introduce în curând calculatoarele, a rezultat faptul că acum, dar mai ales în viitor, un om poate fi considerat alfabetizat numai în cazul când are cunoștințe minimale de informatică și deprinderi practice de utilizare a calculatorului (personal);

— de aici a reieșit necesitatea obiectivă a introducerii informaticii în activitatea instructiv-educativă din școli și în afara ei, ținându-se seama în permanență de particularitățile culturale și social-economice ale fiecărei țări; aceasta presupune dotarea corespunzătoare a școlilor, cercurilor și cluburilor, pregătirea cadrelor și instructorilor, experimentarea și validarea unor programe de instruire teoretică și practică a copiilor, realizarea unor studii și cercetări interdisciplinare, etc.

— educația nu trebuie schimbată în totalitate, fiind necesară doar efectuarea unor modificări corespunzătoare pentru utilizarea optimă a calculatorului în scopul creșterii eficienței procesului de instruire și educație, precum și completarea sistemului cunoștințelor de bază.

În final, s-a tras concluzia că, în viitor, studiul disciplinelor fundamentale (cele clasice) și informaticii (ca o nouă disciplină fundamentală), va avea un rol hotărâtor în formarea cadrelor necesare economiilor naționale.

#### 15.6.2. Moduri evoluat de utilizare a calculatoarelor și microcalculatoarelor personale în învățămînt

În figura 15.11 sînt prezentate sugestiv cîteva din modurile de abordare a procesului de predare/învățare, de la lecția convențională de tip prelegere, pînă la „îndrumătorul electronic” cu sistem video interactiv.

Evoluția materialelor, echipamentelor și tehnicilor utilizate ca suport al lecțiilor ori cursului predat este ilustrată în figura 15.12, prin drumul parcurs de la lecția la tablă, pînă la sistemul video interactiv controlat de microcalculatorul personal.

Dar pentru cei mai mulți, „calculatorul în învățămînt și educație” înseamnă încă, activitatea individuală a unui elev, student sau operator oarecare la un calculator personal, care asistă și completează pentru un timp dat activitatea profesorului, examinatorului sau îndrumătorului.

Dezvoltarea deosebită a domeniului înștruirii asistate de calculator implică utilizarea calculatorului într-o mare varietate de moduri noi, dintre care menționăm cîteva:

- 1) „tablă electronică” ca instrument al profesorului;
- 2) îndrumător (profesor-ghid) electronic pentru elevi și studenți;
- 3) procesor de texte pentru pregătirea rapoartelor, proiectelor și chiar a materialelor didactice, suport al cursului sau cercului de studiu;
- 4) sistem de înmagazinare și regăsire de date (uneori cu facilități de comunicații).
- 5) instrument-asistent pentru o gamă largă de activități (calcul științifice, analiză statistică, prelucrare de tabele, proiectare automată, etc.)



6) sistem pentru conducerea activităților experimentale, sau pentru simularea unor experiențe și legi ale naturii foarte greu sau imposibil de reprodus în laborator.

Trebuie menționat că în general, s-a acordat cel puțin în trecut, o atenție mult mai mare predării *despre* calculatoare, comparativ cu predarea cu *ajutorul* calculatorului în alte domenii.

Progresele recente atât pentru echipamente cât și pentru programe, facilitează introducerea largă și eficientă a calculatoarelor personale și microcalcu-

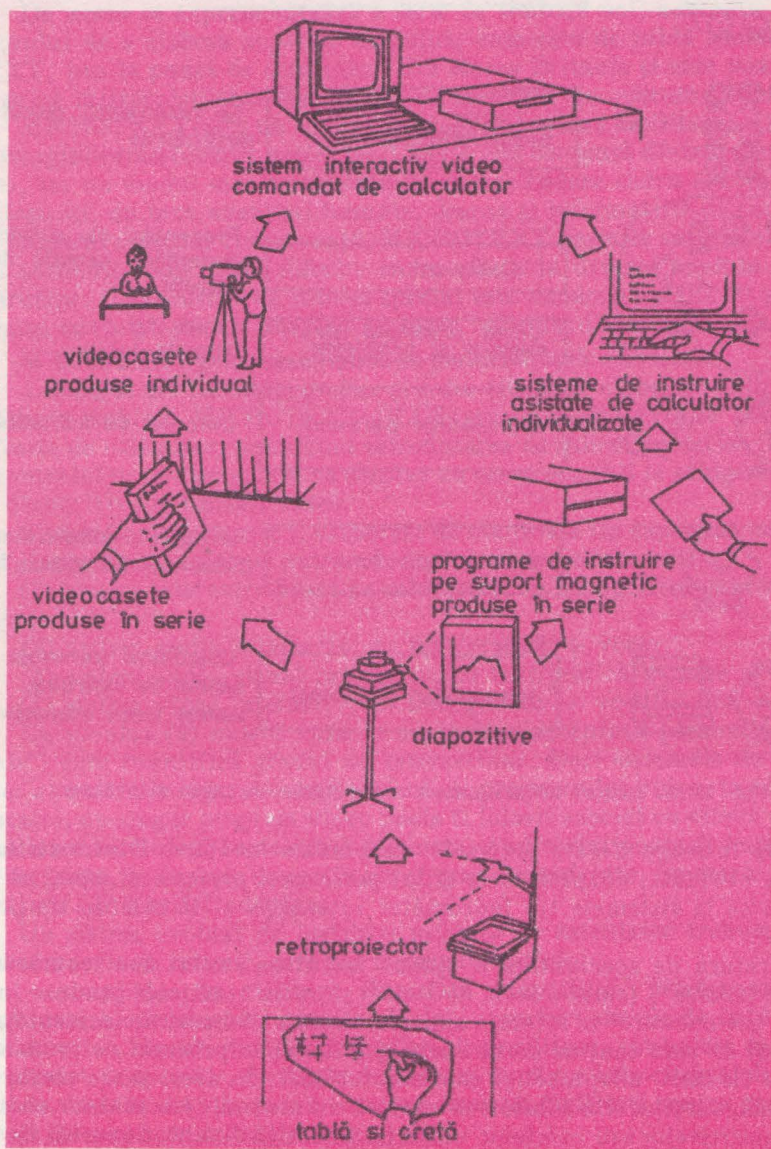


Fig. 15.11. Evoluția suportului de curs

## 15. C.P. ÎN ÎNVĂȚĂMÎNT



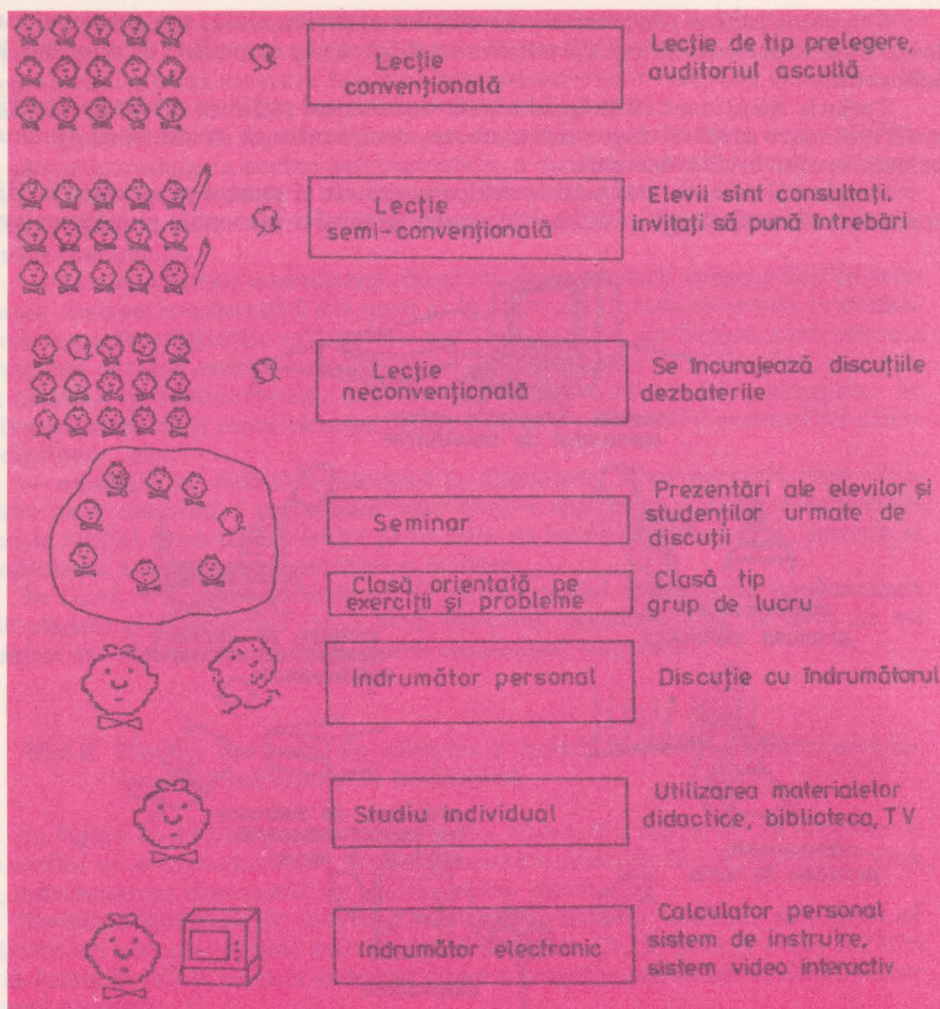


Fig. 15.12. Moduri de desfășurare a proceselor de predare-învățare

latoarelor în întreaga sferă a proceselor de predare-învățare. Dintre sistemele de instruire evolute, adaptabile la stilul persoanei, profesor și elev, ne vom referi mai ales la sistemele tip îndrumător interactiv și la cele de tip „tablă electronică”.

În literatura de specialitate a apărut recent un cuvînt nou „courseware” compus din „course”, (curs, o serie de lecții) și „ware” (articole produse industrial). Prin „courseware” se înțelege suportul de curs elaborat de lectorul cursului. Suportul de curs conține materialele ajutătoare în procesul de predare/învățare, este flexibil, adaptabil și mereu actualizat. În acest sens, acesta este opus de pildă, cărților din bibliotecă care au un conținut fix și chiar cursului scris și diaporitivelor sau casetelor video cu tematică fixată. Suportul de curs elaborat cu ajutorul calculatorului are avantajul că poate fi actualizat rapid, adaptat și multiplicat direct la imprimantă.

## VII. C.P. ÎN ÎNVĂȚĂMÎNT ȘI EDUCAȚIE



Mai mult chiar, discuțiile în clasă obținute interactiv în timpul unei lecții, pot fi transferate direct programului de studiu cu „îndrumător electronic”. Astfel de programe trebuie să devină parte integrantă a cursului și să fie scrise în stilul personal al lectorului.

Oricum, unii elevi învață mai bine dacă revin de câteva ori asupra unui punct în dialogul cu calculatorul, decât să-și dezvăluie neștiința în fața lectorului. Pe de altă parte, un bun sistem de instruire asistată de calculator permite înregistrarea, monitorizarea performanțelor elevilor.

De aceea se experimentează o serie de limbaje noi, adecvate sarcinii profesorului de a elabora suportul cursului, cum este de pildă limbajul MICROTEXT, pus la punct de cercetători din Marea Britanie și care poate fi utilizat pe orice microcalculator compatibil cu sistemul de operare CP/M.

MICROTEXT este un limbaj de programare care facilitează dialogul bazat pe imagini-ecran (cadre), astfel încât lecția este compusă dintr-o secvență de module, iar un modul este o secvență de cadre, fiecare cadru conținând mesajele dialogului, un text, eventual formule, desene și informațiile de control, respectiv numărul cadrului din secvență. MICROTEXT permite salturile condiționate sau necondiționate între diferite cadre aflate în diverse module și utilizează subrutine. O subrutină de interes special este denumită „HELP” și la tastarea acestui cuvânt, apare o imagine cu toate informațiile ajutătoare necesare.

MICROTEXT generează fișiere cu performanțele elevilor.

Limbajul are facilități grafice și color (inclusiv caractere și linii de text cu înălțime dublă). MICROTEXT PLUS are facilități extinse pentru comenzi definite de utilizator, permițând introducerea de subrutine scrise în cod mașină și controlul unor dispozitive ca display cu ecran sensibil la atingere, proiectoare de diapozitive și unități video.

Sistemele de instruire și elaborare de materiale tip suport de curs asistat de calculator, sînt bazate pe patru niveluri de sisteme interactive:

(1) sistem interactiv bazat pe calculator, cu text/grafică generate de calculator;

(2) sistem audio interactiv, cu text/grafică + voce;

(3) sistem audio-vizual interactiv, cu text/grafică + voce + diapozitive;

(4) sistem video interactiv, cu text/grafică + voce + dia + video.

La nivelul (4) există chiar o scală larg acceptată (denumită „Nebraska Scale”) care cuantifică gradul de interacțiune. Deși sistemele de nivel (4) au acces aleatoriu la segmente particulare ale programului video, la capătul de sus al scalei, un singur program video poate fi folosit asociat cu câteva programe de calculator care oferă comentarii distincte. Auditoriul poate fi împărțit în mai multe grupe, în funcție de calificare și grad de instruire, procesele de predare/-învățare fiind astfel adaptate.

Un sistem evoluat de instruire asistată de calculator trebuie să îndeplinească o serie de criterii, care pot fi grupate astfel:

— simplitate în utilizare (interfață adaptabilă, prietenoasă, facilități de ghidare a operatorului numite „Help”, structuri simple de liste de posibilități numite „Menu”, facilități grafice și de editare, simboluri matematice și alte caractere speciale, facilități de căutare după cuvinte-cheie și de manipulare de liste, testare simplă, documentație adecvată, facilități pentru generarea unui suport de curs ușor de utilizat, etc.);

— independență de restricții (compatibilitate cu alte echipamente, dispozitive și programe);



— înregistrarea performanțelor (înregistrarea răspunsurilor date, a punctajului obținut, timpul în care s-a răspuns, editarea unei analize asupra dificultăților ivite în procesul de învățare, facilitatea de „cutie poștală” în care elevii pot înregistra mesajele lor cu sugestii pentru îndrumător sau profesor);

— economicitate.

Asigurarea completă a caracteristicilor pentru unele echipamente noi, de pildă „tabla electronică”, a fost posibilă numai relativ recent, aceasta implicând:

— proiectarea pe ecran mare, pe care să se poată citi textul și grafica, la lumina mediului ambiant;

— un control simplu al afișajului;

— interactivitate.

Dar dintre noile concepte implementate, cel de utilizare intuitivă a calculatorului pare a fi deosebit de atractiv. Astfel a apărut o nouă metodă de manipulare a datelor și un nou mediu denumit WIMP (Window-fereastră, Icon-image/simbol iconic, Mouse-penel, Pointer-cursor).

Mouse, penelul, este de fapt o mică cutie ușor manevrabilă pe o suprafață plană, cutie care are o bilă pentru trasare la bază și unul sau mai multe butoane deasupra. Cutia este conectată printr-un cablu la calculator și amintește de un șoricel cu coada lungă, iar una din funcțiile sale principale este de trasator pe ecran sau de penel. Deplasarea manuală a acestui dispozitiv într-un plan orizontal, face ca un mic semn pe ecran numit cursor să se deplaseze corespunzător pe ecran.

Când se obține poziția dorită a cursorului pe ecran, butonul penelului poate fi utilizat pentru a iniția o acțiune, deschizând sau închizând un fișier de date, sau ștergând ecranul.

Micile simboluri-image dispuse pe marginea ecranului, laturile de jos și din stînga, pot reprezenta diferite părți ale sistemului, ca de pildă fișiere de disc, documente, „scule” pentru pictat, creion, gumă, etc. Deplasarea cursorului la una din aceste micro-imagini și acționarea butoanelor penelului, reprezintă de fapt chiar conceptul central al mediului WIMP. O simplă apăsare a butonului penelului poate selecta un fișier, astfel încît într-o zonă a ecranului se va afișa o listă de posibilități (menu).

Ținînd apăsat butonul și deplasînd penelul la elementul dorit din listă, va fi selectat acel element. Interacțiunea penel/cursor poate fi folosită de asemenea pentru a muta o micro-image între două zone ale ecranului. De exemplu, un fișier poate fi șters mutînd imaginea sa la aceea a coșului de deșeurii, fără a memora și tasta vreo linie de text!

Se pot „decupa” părți din imaginea-ecran, care se pot „salva” și apoi „readuce” printr-o simplă apăsare de buton.

Copierea unui fișier de pe un disc pe altul se obține prin mutarea imaginii asociate fișierului, traversînd ecranul dintr-o zonă (fereastră) în alta.

Pot fi activate simultan pe ecran mai multe zone, corespunzător unor aplicații diverse („multiwindows”).

Textul afișat într-o zonă din ecran poate fi decupat și trecut în altă zonă, permițînd astfel un transfer de date între pachete de programe total diferite.

Dintre sistemele WIMP, cele mai cunoscute și evolute sînt cele implementate pe calculatoarele personale IBM PC și Apple Macintosh. Sistemele WIMP îi încurajează pe începători să facă primii pași mai degrabă prin experimentări, prin manipularea penelului, decît prin consultarea unor manuale extinse. Aceștia învață repede să folosească noul Letraset electronic, să traseze grafice, etc.



### 16.1. Direcții, concluzii și programe ale primului congres internațional UNESCO „Educația și informatica”

Primul congres internațional „Educația și informatica” organizat de UNESCO a avut loc între 12–21 aprilie 1989 la Paris, fiind precedat de multe alte manifestări internaționale, corelate cu evoluția, în ultimele două decenii, a introducerii în învățământ și educație a calculatoarelor electronice.

Pentru a clarifica numeroase aspecte ale *informaticii ca obiect de studiu*, ale utilizării calculatoarelor ca *noi mijloace de instruire*, ale experiențelor dintr-o serie de țări în *strategiile și programele implementate*, cum și ale căilor de întărire a *cooperării internaționale* vom reda realizări, problematici și recomandări legate de pregătirea și desfășurarea acestui congres.

Directorul general al UNESCO, Federico Mayor a sintetizat în cuvîntul de deschidere a congresului problemele conexiunilor dintre educație — factor fundamental al dezvoltării — și noile tehnologii ale informaticii — comunicațiilor, reflectînd rolul profund revoluționar al sistemelor informatice descentralizate și larg răspîndite nu numai în educație ci în întregul nostru mod de viață.

Domnul Federico Mayor a arătat că numeroase probleme importante pentru fiecare și pentru UNESCO s-au pus și se vor dezbate în congres, ca de exemplu: *Ce efect au calculatoarele asupra vieții copiilor și părinților, atît în munca lor cît și în activitățile recreative? Cît costă echipamentele și programele, instruirea și conducerea unui învățământ informatizat? Cum să asigurăm cele mai bune utilizări ale acestei tehnologii în sistemele noastre de învățămînt? Cum să integram subiecții izolați sau handicapați în structurile educaționale tradiționale? Poate oare informatica să asigure o creștere a numărului absolvenților și o ridicare a pregătirii în școlile primare și secundare? Ca prime răspunsuri la acestea s-au selectat cîteva aspecte reflectînd avantajele și dezavantajele utilizării și viitorului informaticii în învățămînt:*

— Opiniile contrastante asupra oportunității utilizării informaticii în educație în diferite țări și centre care se înscriu pe o paletă „de la lipsa de interes la entuziasm”. În stadiul inițial de informatizare, în multe țări s-a apreciat că se încearcă a *substitui profesorii prin mașini*; experiența a arătat apoi că pentru eficiența învățămîntului asistat de calculator, profesorii trebuie să fie *puternic instruiți*, această cerință obligatorie ca și faptul că software-ul disponibil era (este) insuficient sau inadecvat a făcut ca în unele țări acest tip de învățămînt să fie neglijat; dar, deasupra acestor factori tehnologici, rămîne întrebarea dacă informatica poate ajuta la îmbunătățirea calității educației. Cităm patru aspecte benefice:

1° — Crearea unor *vaste și ieftine resurse* care produc schimbări cantitative și, mai ales, calitative în cunoștințele și informațiile disponibile copiilor / elevilor, profesorilor, părinților.

2° — *Noi metode de educație*, cu noul mijloc important al *autoinstruirii*.

3° — *Creșterea numărului de tineri* (și nu numai al lor), care s-au familiarizat singuri cu noile tehnici.

4° — *Convingerea crescîndă* că se deschid multe căi de rezolvare a problemelor pornind de la cunoașterea „proaspătă” a informațiilor.

Desigur, ignorăm *dezavantajele* utilizării informaticii în educație: *costul hardware-ului și software-ului; oboseala* cauzată de utilizarea excesivă a calculatoarelor; scăderea posibilă a aptitudinilor mentale pentru operații pe care acum le fac mașinile; *lipsa de flexibilitate* (frecvent dar nu inevitabilă) a metodelor de învățare impregnate cu actualul software educațional. Este o cale lungă pentru calculatoare în devenirea lor ca instrumente flexibile



pentru instructori, asigurând atât „izolări“ în studiu ale copiilor cît și creșterea atenției lor față de profesori. De asemenea, anumite subiecte, cum ar fi învățarea limbilor sînt, prin ele însele, mai puțin adaptate decît altele pentru instruirea asistată de calculator.

— Multe din situațiile prezentate se referă la țările industrializate, în țările în curs de dezvoltare aparînd ca esențială introducerea tehnologiei informaticii în condiții de *resurse limitate*; cerințele costurilor reduse și a îmbunătățirii performanțelor economice nu s-au prea realizat. Pe de altă parte, informatica este — în parte — responsabilă de *discrepanța* dintre țările industrializate și cele în curs de dezvoltare. Apar probleme specifice ca: *buna informare* asupra noilor tehnici; *transferul de tehnologii*, în cît mai echitabile condiții pentru țările în curs de dezvoltare și pentru lumea a treia; este necesar un sprijin real al producătorilor de echipamente și programe din țările industrializate.

— O cale eficientă pentru toate țările în vederea „expansiunii“ și îmbunătățirii aplicațiilor informaticii în educație este *crearea unor organisme sau centre specializate* care să observe, să urmărească, să evalueze și să răspundă cît mai larg cu putință progresul în informatică, atât din punct de vedere tehnic cît și educațional. Cooperarea în acest domeniu se încurajează prin *schimburi de informații și de experți*, prin implementarea de *proiecte comune* pentru producerea de hardware și software utilizabil pentru instruirea profesorilor și educatorilor, prin stabilirea unor *criterii internaționale* pentru selectarea, evaluarea și dezvoltarea de software educațional.

Ce putem afirma despre viitorul informaticii în educație? Trei *tendențe* sînt de menționate:

- *autoinstruirea*;
- *activitatea în locuințe*;
- *instruirea de la distanță*

Educatorii de azi vor trebui să fie la curent tot timpul cu inovările tehnologice rapide și să apeleze atât la *psihologie*, *sociologie* cît și la *știința calculatoarelor*. UNESCO îi va sprijini.

Manifestările internaționale, ca acest prim Congres, permit schimburi largi de vederi între educatori reprezentanți ai lumii industriale, editori, cercetători, decidenți. De asemenea se evaluează situația în lume, se descoperă tendințele de dezvoltare și cooperare internațională în utilizarea informaticii și tehnologiei comunicațiilor în domeniul educației.

Programul Congresului include 5 teme majore:

- 1° — *Evaluarea prezentului și obiectivele de atins*
- 2° — *Strategii naționale și posibilități de extindere internațională.*
- 3° — *Cooperarea cu industria*
- 4° — *Aplicațiile informaticii ca mediu de instruire și educare*
- 5° — *Perspective de viitor*

îndreptate toate spre extinderea cooperării internaționale.

[Dl. Federico Mayor a menționat rolul în pregătirea congresului al domnului Robert Chapuis, secretarul de stat pentru educație tehnică al Franței, a d-lui F. Peregudov, primul vicepreședinte al Comitetului de stat pentru educația maselor din U.R.S.S. și a d-lui André Danzin, președintele Congresului și al Programului informatic interguvernamental. Franța și U.R.S.S. au fost statele care au inițiat rezoluțiile ce au dus la organizarea congresului].

La închiderea congresului s-au prezentat concluzii generale sintetice.

*Concluziile generale* ale primului Congres internațional „Educație și informatică“ al UNESCO, aprilie 1989, Paris, (la care au participat peste 500 reprezentanți din 93 țări și 29 organizații internaționale) sînt, în rezumat, următoarele:

1. Informatica este capabilă să realizeze eficacitatea internă și externă a sistemelor educaționale, cum rezultă din aplicațiile limitate dar semnificative și din tehnologiile noi ale informației realizate în ultimii cîțiva ani.

2. Introducerea noilor tehnologii informatice în educație (NTIE) riscă să creeze disparități din domeniu, în interiorul țărilor și între ele, dacă nu se întreprind acțiuni energice cu măsuri specifice la nivel național și internațional pentru a evita o atare evoluție.

3. Tehnologiile informatice noi joacă un rol important în aspecte majore ale dezvoltării socio-economice și culturale; cooperarea internațională trebuie întărită, trebuie sprijinite țările mai puțin dezvoltate, pentru a utiliza mai bine aceste mijloace în contextul propriu al fiecăreia.

4. Aceste tehnici trebuie să facă parte din cultura accesibilă ansamblului populației.

5. NTIE au multiple roluri, nu numai ca unelte pedagogice dar și ca o abordare și o cultură noi, permițînd un dialog concret în cadrul învățămîntului și practicii gestiunii informației și al dezvoltării accelerate a societății.



6. Sînt necesari profesioniști foarte înalt calificați, cu competențe atât tehnologice cît și în științele cunoașterii.

7. Este important să reorientăm și să formăm progresiv elevii și alte persoane din domeniul educației, pentru a le permite să exploatzeze NTIE.

8. Recunoaștem necesitatea continuării cercetărilor asupra inerției noilor tehnologii ca unelte ce ameliorează desfășurarea procesului pedagogic.

9. Notăm contribuția pozitivă a întreprinderilor particulare și publice producătoare de hardware și software care servesc cauza dezvoltării de NTIE.

10. Vom recenza domeniile învățămîntului în care NTIE pot crește cel mai mult randamentul și pot maximiza fructele practicii.

11. Vom întări cooperarea internațională pentru a facilita inovarea, experimentarea și cercetarea privind aplicațiile pedagogice ale NTIE.

12. Este necesar, pentru a stimula integrarea NTIE, ca industria informaticii să poată să se plaseze într-o perspectivă pe termen lung, pentru a determina domeniile de interes comun cu sectorul educației și de a aduce la dezvoltarea acestui domeniu aportul necesar pentru atingerea concretă a obiectivelor sale.

13. Ne propunem să remediem inegalitățile dintre țări în ce privește dezvoltarea NTIE prin creșterea cooperării internaționale prin diverse mijloace:

- aparținînd identităților, culturile și limbile naționale;
- schimbînd informații în cadrul reuniunilor, seminariilor, cursurilor, vizitelor, congreselor și prin alte mijloace de comunicare „față în față“;

- întreprinzînd proiecte și cercetări pentru strategia de utilizare a NTIE, utile pentru toate țările, cercetînd și posibilitățile de a asigura transferabilitatea, complementaritatea, modularitatea și compatibilitatea programelor, echipamentelor, a sistemelor și rețelelor de comunicații informatice utilizabile în educație și formare;

- favorizînd circulația internațională a informațiilor prin rețelele informatice internaționale și prin alte mijloace de comunicație;

- ajutînd țările mai puțin favorizate și pe cele în curs de dezvoltare pentru achiziția, utilizarea și producția de echipamente, facilitîndu-le și accesul la software de aplicații didactice prin licențe și cooperări, grație dezvoltării capacităților naționale de concepere a programelor de aplicații didactice conforme cu necesitățile și cultura locală.

14. Apelăm la indivizi, colectivități, întreprinderi, guverne pentru a da un nou elan concertațiilor atât de necesare în domeniu și a asigura planuri și programe de aplicație suficiente pentru toate nevoile exprimate.

15. Apelăm la UNESCO, la ONU, la Organizația mondială a muncii, la ONUDI, la Programul ONU pentru dezvoltare, la Banca mondială, la băncile regionale de dezvoltare și la alte organizații interguvernamentale și guvernamentale pentru a susține aplicarea judicioasă a NTIE și a contribui la umplerea „gropii“ existente în domeniu între țările dezvoltate și cele în curs de dezvoltare.

## PROGRAMUL COMENTAT AL CONGRESULUI UNESCO „EDUCAȚIE ȘI INFORMATICĂ”

*Cuvîntul directorului general UNESCO*

*Introducere: Informatica, societatea și dezvoltarea: ce informatică și pentru ce dezvoltare (informații de organizare, regulamente ale reuniunilor UNESCO)?*

**TEMA 1: SITUAȚIA ACTUALĂ ȘI OBIECTIVELE** (răspunsul statelor membre ale UNESCO la chestionarul prealabil.

*Masă rotundă: Obiectivele introducerii informaticii bazate pe diferite experiențe naționale.*

**Tema 1.1. Aplicații ale informaticii în cadrul educației permanente** (o societate se dezvoltă în măsura în care este capabilă să învețe: educația a încetat să fie o experiență limitată la o singură perioadă a vieții; sînt necesare legături între contextele formale și neformale în care se desfășoară educația; ce este inițierea în informatică pentru marele public? Ce relații există în cadrul întreprinderilor, societăților?

**Tema 1.2. Informatica, egalizarea accesului pentru diferite grupe și egalizarea șanselor școlare** (între altele cazul particular al femeilor, handicapaților, izolaților, bolnavilor).



Tema 1.3. *Informatica, transformarea locului de muncă și a formației profesionale* (politicile educative trebuie să țină seama de natura mereu în schimbare a muncii, altfel oamenii formați riscă să-și dezvolte rapid sentimente de frustrare și devin într-un anumit fel niște nealfabetizați, lipsiți de lucru).

Tema 1.4. *Noi raporturi între partenerii din procesul educațional* (instituții, întreprinderi, colectivități, mișcări asociative și cooperative, familie, care sînt consecințele locului crescînd acordat informaticii).

Tema 1.5. *Restricții economice și financiare, evaluarea costurilor investițiilor și funcționării* (echipament, programe, formare, cercetare, dezvoltare, evaluare... care sînt resursele naționale necesare pentru introducerea informaticii în educație și care ar fi raportul cost / beneficiu al unei asemenea întreprize).

Tema 1.6. *Introducerea informaticii în educație la nivel național*: obiective, oportunități, strategii (unele țări au rezultate, altele n-au experiență; care ar fi nevoile și care ar constitui justificări de lansare a unor asemenea planuri).

**TEMA 2: STRATEGII NAȚIONALE ȘI LĂRGIREA LOR PE PLAN INTERNAȚIONAL** (noile sisteme de învățare; ca mijloc de planificare, gestiune și de control al învățămîntului într-un mediu în schimbare); se pot ameliora performanțele sistemului educativ? productivitatea poate crește și eficacitatea poate fi măsurată la 4 nivele: sistemul național, circumscripția educativă, grupul școlar, unitatea pedagogică; strategii pentru țările în curs de dezvoltare, cooperatii între țări cu probleme similare sau vecine, sau aflate la stadii de dezvoltare comparabile, mai ales în domeniul software-ului, programelor didactice și formarea echipelor integrate.

Tema 2.1. *Formarea profesorilor și instructorilor* (formarea inițială, cursuri de perfecționare, conținut și metode)

Tema 2.2. *Producția, difuzarea și transferul de mijloace informatice* (echipamente, programe generale și educaționale, baze de date și rețele și adaptarea lor la contexte diferite; diferite strategii, rolul sectorului public și a celui particular, standardizării, evaluări; programe-naționale, transfer între țări; posibilități de adaptare.

Tema 2.3. *Aplicarea informaticii la gestiunea proceselor educative* (la diferite nivele instituționale) ca și a formării „gestionarilor” (contabilitate, dosare elevi, note examen, biblioteci, materiale didactice; gestiunea personalului, construcțiilor, bugetelor, perspectivelor; problema costuri, aplicații noi mai ieftine).

**TEMA 3: CONCERTAREA CU INDUSTRIA.** *Dezbateri la SICOB*: educatori-industriași-editori, relații cu autoritățile educative, contracte; software-societățile de servicii (etc. editori), traducerea programelor, participarea industriei la formare și la întreținere, complemente ale ofertelor de echipamente; adaptarea echipamentelor programelor pentru țări în curs de dezvoltare; convivialitate, fiabilitatea soft și hard.

**TEMA 4: PRACTICI** (— ca studiu transdisciplinar; ca mijloc de transmitere a informațiilor pentru utilizare în diverse configurații educative; interacțiunea între informatică și conținutul cunoștințelor, fenomen important, cu noi perspective în știința cunoașterii.

Tema 4.1. *Informatica drept mijloc pentru noi situații pedagogice* (organizarea învățămîntului, individualizarea / personalizarea și reformularea raporturilor profesor-elev; se învață mai bine și mai repede folosindu-se și mijloacele informatice? experiențe în curs, evaluări.

Tema 4.2. *Software de bază și de aplicații didactice* (baze de date, simulare, modelare, lexicografice; posibilități și limite actuale; software-echipament. Nevoi imense și permanente de concepere și producție de software educațional).

Tema 4.3. *Informatica și dezvoltarea învățămîntului post secundar și universitar* (cum?; noile modalități de învățare la distanță).

**TEMA 5: PERSPECTIVE** (perspectivă tehnologică — a cincea generație, telematică, inteligență artificială; perspectivă socio-culturală — cunoașterea, valoarea comercială; care sînt tendințele actuale în materie de acces individual și colectiv la informație și cunoaștere? echitate la nivel național și internațional; consecințe asupra proceselor mentale ale generalizării informaticii în societate; consecințe pentru obiective și conținuturi educative pornind de la metode și tehnici.

Tema 5.1.: *Cercetarea asupra informaticii în educație* (situație, dezvoltarea proiectelor, cooperarea internațională; cum să întărim aspectele pedagogice ale cercetării? cum să identificăm mai bine și să promovăm orientarea asupra temelor de cercetare la nivel național și internațional?



**Tema 5.2.: Impactul informaticii asupra limbilor și culturii naționale în diversitatea lor** (limbaj artificial și natural; „alfabetizare informatică”, probleme puse de proliferarea mijloacelor și aplicațiilor informatice în societate.

**Tema 5.3.: Pentru o dinamică a cooperării internaționale.** Utilitatea evaluărilor retrospective; schimburi de experiență între situații similare; studii comparative; cedarea de experiență în materie de formare a studenților, de elaborare a politicilor / planurilor / strategiilor / programelor, de aplicații pedagogice și de producție de software; rațiuni tehnice cu partenerul cel mai apropiat.

*Inchidere. Ce politică și ce cooperare internațională?*

## 16.2. O programă analitică americană pentru „alfabetizarea” informatică\*

Am consultat numeroase materiale bibliografice pregătitoare și din conținutul propriu zis al primului *Congres internațional UNESCO, "Educație și Învățămînt"*, Paris, aprilie 1989, materialele *Conferinței miniștrilor educației din statele membre ale Europei*, Paris, septembrie 1988, *"Informatica în educație"*, o serie de studii relativ independente pentru țări și grupe de țări din periodicele trimestriale UNESCO *"Perspectives"* 1987, cercetările expuse în conferințele *„Copiii în epoca informației"*, Varna, Bulgaria, 1985, 1987, 1989 ș.a..

Drept model exemplificativ de programă analitică pentru „alfabetizarea” informatică am reținut, însă, extrase adaptate dintr-o carte americană, apărută în 1983, sub coordonarea lui Beverly Hunter *„My students use computers; learning activities for computer literacy"*, editată de Prentice Hall Company, în S.U.A.

La *elaborarea* sa au participat peste două sute de specialiști cu pregătiri diferite (echipă multidisciplinară), cooperînd, la îndemnul conducerii Organizației de cercetare a resurselor umane (HRRO), pe baza unor experimentări și testări ample în școala publică din Montgomery County, Maryland și în alte școli pilot, avizate critic de un grup național de profesori consultanți de la numeroase departamente, școli, consilii, firme de calculatoare, edituri, fundații. *Experimentările* au început în 1980 pe computere APPLE-PET (40 calculatoare personale în 165 clase din școli elementare, gimnaziale și liceale). Azi (după 3 de ani) toate clasele din experiment au acces la calculatoare personale (400 calculatoare personale performante), cumpărate din fonduri centrale și particulare. Profesorii au fost pregătiți, în mare măsură, prin cursuri de 45 de ore în ultimii doi ani.

Scopul acestei programe analitice este să dezvolte o „alfabetizare” informatică universală, necesară tuturor cetățenilor pentru a supraviețui într-o societate care depinde de tehnologiile informatice.

**Definiție.** „Alfabetizarea” care se realizează în 8 etape sau stagii temporale (grade, clase), își propune a asigura *abilitatea* de a utiliza programe de calculator adecvate ce asistă instruirea, de a manipula informații și de a rezolva informatic probleme, ca și *abilitatea* de a raționa judicios asupra problemelor sociale și etice care implică sisteme de comunicații și calculatoare.

*Programa* este organizată pe 6 grupe tematice, fiecare grupă incluzînd o serie de obiective care trebuie asigurate în fiecare din cele 8 stagii (grade), grupate câte două (1–2), (3–4), (5–6), (7–8).

Cele 6 grupe tematice (aceleași pentru cele 8 stagii / clase) sînt:

- I. Utilizarea și dezvoltarea procedurilor
- II. Utilizarea unor programe pentru calculator
- III. Concepte fundamentale despre calculatoare
- IV. Aplicațiile calculatoarelor
- V. Impactul calculatoarelor asupra societății
- VI. Scrierea programelor de calculator

---

\* Beverly Hunter, *My students use computers*, Reston Publishing Company inc. Virginia, S.U.A., 1983.



Conținutul acestor grupe tematice se modifică în funcție de gradele / stagiile (1-2), (3-4), (5-6), (7-8).

### Grupa tematică I. Utilizarea și dezvoltarea procedurilor

O procedură este o descriere pas cu pas a ceea ce trebuie să facem; o procedură este un set specific de instrucțiuni care trebuie realizate în exact secvențele specificate. Procedura include algoritmi de rezolvare. Oamenii pot face multe lucruri fără a fi conștienți de o procedură, care manevrează un mare volum de informații în căi sistematice în scopul rezolvării problemelor. Elevii vor învăța conceptele relative la proceduri prin:

a) *Descrierea unei activități familiare în termenii procedurilor*, în mod complet și în ordine corectă a pașilor.

b) *Utilizarea procedurilor pentru rezolvarea problemelor* în matematică, studii sociale, științe, artă și viața de fiecare zi.

c) *Dezvoltarea procedurilor pentru a rezolva noi probleme*, elevul deprinzând mijlocul intelectual esențial de a diseca (sparge) o problemă în subprobleme inteligibile și mijlocii, de a dezvolta metode pas cu pas pentru obținerea soluției. Elevii trebuie să înceapă cu conceptul de procedură înainte de a învăța programarea calculatorului, care i-ar putea distra prin sintaxa limbajelor de programare. Ei trebuie să înțeleagă mai bine cum rezolvă omul problemele și cum, prin proceduri explicite, formale, poate lucra calculatorul. În clase, elevii sînt deprinși a urmări un set de instrucțiuni pentru a realiza un set de instrucțiuni mai detaliat, pentru a le testa și a le corecta. Elevii trebuie să exprime apoi aceste proceduri în forma unui program de calculator. În cazul nostru s-a adoptat limbajul de programare de nivel înalt LOGO.

Există diferențe între obiectivele grupei tematice în cele 8 stagii (luate două câte două 1-2, 3-4, 5-6, 7-8).

În stagiile 1-2 elevii învață,

- 1° — să execute o procedură prin instrucțiuni pas-cu-pas pentru o sarcină familiară;
- 2° — să modifice o procedură pentru a rezolva o sarcină similară, nouă;
- 3° — să demonstreze diferite proceduri utilizabile pentru aceleași ieșiri;
- 4° — să descopere și să corecteze erorile (inclusiv în secvențe greșite a instrucțiunilor) pentru proceduri cu pînă la 10 pași și care nu conțin ramuri de luarea deciziilor;
- 5° — să descrie procedurile pentru realizarea unei sarcini;
- 6° — ajută la dezvoltarea unei proceduri scrise, pentru o sarcină familiară.

În stagiile 3-4:

- 7° — ajută la dezvoltarea unei proceduri care include repetiții, decizii, intrări variabile.
- 8° — găsesc și corectează erori în proceduri care includ repetiții, decizii și intrări variabile;
- 9° — dezvoltă independent o procedură care poate fi executată de o persoană care are o sarcină — și îi dă un nume —, demonstrînd că prin procedură se ating rezultatele scontate;

În stagiile (clasele) 5-6

- 10° — identifică anumite diferențe între o procedură executată de om și una de calculator;
- 11° — „sparg” o problemă în subprobleme; planifică un set structurat de proceduri și subproceduri pentru a rezolva problema;
- 12° — dezvoltă o procedură prin organizarea datelor pe o anumită cale, necesară pentru analiza și rezolvarea problemei; prezintă o soluție (de ex. pentru ordonarea alfabetică sau numerică a datelor, scăderea sau adunarea datelor, tabelarea, exprimarea grafică a datelor etc).

În stagiile (clasele) 7-8

- 13° — rezolvă o problemă prin diferite metode cu creionul și hîrtia, cu un calculator, cu un program de calculator preîmpărit și cu un nou program (punînd accentul pe factori ca volumul și plaja datelor utilizate, prelucrate sau stocate în rezolvarea problemei, dacă problema este cuantificabilă sau nu; dacă există sau nu un alt program de urmat.



## Grupa tematică II. Utilizarea unor programe pentru calculator

Pentru a fi alfabetizați elevii trebuie să utilizeze diferite tipuri de programe pe calculatoare personale. Programele utilizate trebuie să atingă, împreună cu familiarizarea cu o largă varietate de aplicații, trei scopuri principale:

- să ajute în achiziționarea ideilor, conceptelor și atitudinilor prescrise în lecțiile științifice, matematice, studiilor sociale, artelor; programele să fie utilizabile în învățare;
- să ajute și să încurajeze elevii în selectarea programelor utilizabile în sarcinile ce le au, să fie capabili de a citi (și chiar a scrie) documentarea programelor, ca sprijin în selecția și operarea programelor și echipamentelor; acesta este aspectul de „ținere sub control” a calculatorului;
- să învețe elevii să fie responsabili, într-o manieră etică în raport cu echipamentul, informația și programele, care sînt utilizabile de multe persoane.

Multi profesori menționează că utilizarea programelor de calculator constituie un nou nivel de motivație pentru elevi.

În stagiile (clasele) 1—2 se învață:

- 1° — utilizarea programelor simple de autoinstruire și jocuri ca asistînd învățarea matematicii, științei, studiilor sociale, limbajului artelor;
- 2° — dispozitivele de operare (cum sînt tastaturile, calculatoarele, terminalele video, șoricelul, joystick-ul, caseta, discul rigid, discurile flexibile) și comenzile necesare pentru a utiliza programele disponibile (încărcare, rulare, listare și oprire a programului).
- 3° — să recunoască că, pentru a opera, calculatorul trebuie să primească instrucțiuni ordonate și că acestea trebuie să fie condiționate în program;
- 4° — urmărind legile și regulile etice de utilizare a programelor și echipamentelor, elevii să protejeze echipamentele împotriva defectărilor, să nu distrugă datele sau programele altor persoane și să nu întrerupă intenționat operarea în sistemul de programe; să respecte regulile de planificare și distribuie a echipamentelor și programelor.

În stagiile (clasele) 3—4:

- 5° — să citească și înțeleagă documentația unui program (nume program, scopul programului, cum se utilizează programul, datele și echipamentele necesare pentru utilizarea programului);
- 6° — să utilizeze documentația pentru utilizarea unui program adaptat unei sarcini date și să opereze cu programul, fără asistența profesorului;
- 7° — să selecteze și să utilizeze un program care îl ajută la soluționarea unei probleme atribuite.

În stagiile (clasele) 5—6:

- 8° — să utilizeze simularea pe calculator ca asistare în instruire după planul de studiu;
- 9° — să demonstreze înțelegerea efectelor și interacțiunilor unor variabile selectate într-o simulare utilizată; să descrie anumite aspecte ale lumii reale care sînt incluse sau eliminate din simularea utilizată;
- 10° — să regăsească informații selectate dintr-o bază de date informatică, pentru scrierea unei lucrări sau a unui proiect.

În stagiile (clasele) 7—8

- 11° — să utilizeze un program de procesare de texte ca asistent în scrierea, revizuirea și realizarea unei culegeri.

## Grupa tematică III. Concepte fundamentale despre calculatoare

Sînt puține concepte fundamentale despre calculatoare pe care fiecare trebuie să le înțeleagă că necesită instrucțiuni ordonate pentru a opera: că este o mașină cu scopuri generale; că operează foarte rapid; că poate face multe repetări ale aceluiași operații pe exact aceleași căi; poate prelucra un mare volum de date.

Pentru a înțelege aceste caracteristici elevii trebuie să vizualizeze sarcini sau proceduri cu mai mulți pași, cu volum mare de date, rezolvate cu viteze mari; aceste idei permit să se introducă în matematică, științe și studii sociale a unor probleme de complexitate conformă cu a lumii reale.



În stagiile 1—2 (sau cît de repede încep să utilizeze calculatorul) elevii trebuie să învețe:

1° — să recunoască și că pentru operare calculatorul trebuie să primească instrucțiuni ordonate, prin program;

În stagiile 3—4

2° — să recunoască și că un calculator poate să opereze cu diferite programe

În stagiile 5—6:

3° — să dea exemple de sarcini în care viteza calculatorului este necesară sau avantajoasă și să o poată compara cu viteză umană corespunzătoare;

4° — să dea exemple de sarcini care implică multe repetări ale acelorași operații și explicații despre utilitatea calculatorului în acest caz;

5° — să dea exemple de sarcini care necesită prelucrarea unui volum mare de date și explicații despre utilizarea calculatorului în acest caz.

În stagiile 7—8

6° — să aplice înțelegerea conceptelor în multe situații noi.

#### Grupa tematică IV. Aplicațiile calculatoarelor

Calculatoarele sînt utilizate în sute de căi de către indivizi și organizații; aceste utilizări se cheamă „aplicații” sau „sisteme” informatice.

Aceste aplicații presupun oameni care interacționează între ei și cu mașinile; toate operează cu date, stocate în anumite căi, organizate în forme pe care calculatorul să le „înțeleagă”; toate sistemele au *intrări* (instrucțiuni și date) de la diferiți oameni și din diferite locuri; toate sistemele au *ieșiri* (informații și instrucțiuni) de la oameni sau de la alte mașini; oamenii și mașinile din aplicație au anumite mijloace de *comunicare*; trebuie să existe proceduri proiectate care specifică secvența acțiunilor.

În stagiile 1—2

1° — descrierea aplicațiilor din locuințe și vecinătăți (cine și de ce utilizează calculatoare; ce date dau persoanele calculatorului și ce date dau calculatoarele persoanelor; cum transformă persoanele datele în informații).

În stagiile 3—4

2° — se descriu aplicațiile din școli, organizații administrative și comerciale, la fel ca la 1° dar cu noțiunile de *intrări* și *ieșiri*, adăugîndu-se și „pozițiile” oamenilor și mașinilor, comunicațiile în sistem și tipurile de informații în bazele de date informatice;

3° — similaritățile și diferențele între calculatoarele imaginate și cele reale.

În stagiile 5—6

4° — se descriu una sau mai multe căi prin care informația regăsită este utilizată în domenii ca știința, dreptul, educația;

5° — se descriu una sau mai multe aplicații științifice (ce se studiază, ce tipuri de date oferă calculatorul, cum se utilizează datele, ce tipuri de date sînt *intrări*)

În stagiile 7—8

6° — se descriu una sau mai multe căi în care prelucrarea textelor se utilizează în comerț, administrație, jurnalism;

7° — se descriu aplicații în comerț, finanțe, fabricație, transporturi;

8° — se descrie un sistem cu calculator utilizat într-o mare organizație (diferiți utilizatori cu pozițiile lor, tipuri de informații provenite de la ei, sursele de date de intrare în sistem; anumite tipuri de date conținute în baza de date; cum comunică între ele calculatoarele; unele tipuri de programe care rulează în sistem; surse de erori posibile în sistem);

9° — recunoașterea interacțiunilor în sistem între oameni, hardware, proceduri, software, documentație, comunicații, date.



## Grupa tematică V. Impactul calculatoarelor asupra societății

Calculatoarele au puternice efecte asupra societății și individului; se speră efecte benefice; interesează ce fel de efecte apar în raport cu diferite grupuri sau indivizi; elevii trebuie să discearnă avantajele și dezavantajele; importante sînt considerațiile de comportament și din sistemele informatice.

### În stagiile 1-2

- 1° — se urmăresc regulile și legile de comportament (defectări daune, distrugerea unor date sau programe ale altora, copierea neautorizată a programelor ce au copyright, folosirea codului de identificare al unei alte persoane).

În stagiile 3-4 elevii continuă să învețe aceste reguli și rațiunile lor și sînt introduși în diferite tipuri de programe.

### În stagiile 5-6 (în conexiune cu stocarea informațiilor și secțiunea de regăsire)

- 2° — explicarea rațiunilor de restricții și de control al accesului la aplicație (protecție împotriva fraudelor, dreptului particular, erori și baza de date ș.a.);
- 3° — listarea avantajelor și dezavantajelor unei aplicații de regăsire a informației utilizate (de ex. *avantaje*: rapiditate, acces a multor persoane, disponibilitatea unor informații noi; *dezavantaje*: pot fi erori necunoscute în baza de date; se pot utiliza informații în căi ilegale sau necorecte față de alții, se pot pierde informații din cauza formatelor standardizate sau a memoriei limitate)

### În stagiile 7-8

- 4° — se analizează ieșirile calculatoarelor și locația informațiilor pentru analiză (inteligenta artificială, automatizarea, controlul utilităților și facilităților de comunicații, furtul în calculator ș.a.). Analiza include identificarea ieșirilor — a părților în conflict, a pozițiilor părților în conflict, a ieșirilor necesare pentru calificarea faptelor;
- 5° — identificarea și recunoașterea anumiților factori care contribuie sau care sînt suportați de crescînda dependență a noastră față de sistemele informatice (dimensiunile și complexitățile crescînde ale organizațiilor guvernamentale și de afaceri; creșterea nivelului educației; creșterea nivelului de informații generate sau comunicate; reducerea dimensiunilor și costurilor calculatoarelor; creșterea vitezei și a capacității de stocare; îmbunătățiri în căile de comunicare ale oamenilor cu calculatorul; marketingul în industria calculatoarelor);
- 6° — se descriu căi în care oamenii pot fi afectați de „căderile” și greșelile calculatoarelor — erori de calcul, „căderi” în traficul aerian, plăți eronate, erori de diagnostic sau tratament
- 7° — se descriu cerințe ale oamenilor față de industria calculatoarelor (întrări de date prietenoase; interogarea bazelor de date; specificarea cerințelor pentru prelucrarea datelor, asistarea cu calculatorul a echipamentelor de laborator, echipamente de prelucrarea textelor, utilizarea proiectării asistate de calculator);
- 8° — se descriu diferite atribuții legate de industriile calculatoarelor și prelucrării datelor (programator, proiectant și realizator de calculatoare, service-man, analist)

## Grupa tematică VI. Seriarea programelor de calculator

Deoarece tehnologia calculatoarelor și aplicațiile sînt în rapidă schimbare, nu există un consens între tehnologi și educatori privind scopurile, obiectivele și metodele de învățare a programării calculatoarelor pentru toate categoriile de copii și adulți. Cartea citată propune: un set de obiective independente de limbajul de programare; o discuție asupra limbajelor alternative (BASIC, LOGO, PASCAL); planuri de lecții LOGO; bibliografii pentru predare și învățare în cele patru limbaje sugerate; recomandarea ca introducerea în programare să fie o unitate (sau un curs) separată care să integreze și aplicații în matematică, studii sociale, limbajul artelor; integrarea și a altor domenii ce interesează pe elev).

Obiectivele pentru scrierea unor programe de calculator au fost plasate la stagiile (clasele 7-8); dacă la anumite școli se cer scrieri de programe la alte clase se va folosi experiența acumulată în stagiile 7-8. Rațiunea plasării conceperii programelor la stagiile 7-8 este explicată și de: gîndirea procedurală din stagiile 1-6 este preliminară scrierii unor programe



utilizabile, studiul pînă la stagiul 6 a gîndirii procedurale permite succesul scrierii de către toți elevii a programelor în stagiile 7-8; multe școli sînt organizate să includă un curs de programare de abia în clase corespunzătoare cu stagiile 7-8.

#### Stagiile 7-8

- 1° — modificarea unui program ca un sprijin în învățare sau rezolvarea problemelor;
- 2° — translatarea unei proceduri date într-un program utilizînd instrucțiuni convenabile (format de date numerice sau șiruri; acceptarea mutărilor de la tastatură sau alte dispozitive; ieșiri pe display sau videodisplay, pe imprimantă sau pe alte dispozitive, executarea operațiilor logice sau aritmetice; manipularea datelor);
- 3° — scrierea unui program într-o formă și un stil care poate să servească altora pentru găsirea și corectarea erorilor, utilizarea și modificarea programului;
- 4° — testarea și depănarea unui program scris și controlul corectitudinii rezultatelor; date de test, mesaje de eroare și depănarea prin cunoașterea unor rezultate intermediare;
- 5° — urmăriind un set ordonat în planificarea și dezvoltarea unui program să rezolvăm o problemă particulară (definirea; spargerea în subprobleme, prepararea datelor de test; dezvoltînd prin pseudocod sau prin diagrame logice, un set de proceduri și subproceduri pentru rezolvarea problemei; controlul erorilor logice; translatarea procedurilor în codurile limbajelor de programare; editarea și testarea programului; localizarea și corectarea erorilor; pregătirea instrucțiunilor pentru utilizatorii programului; utilizarea programului).

Cartea citată este deosebit de completă, aici nu s-a făcut decît o selectare din programa matricială (grupe tematice / stagii (clase) care este în detaliu completată cu tehnici, studii de caz, anexe etc.

### 16.3. Programul detaliat al disciplinei „Bazele informaticii și tehnicii de calcul” pentru învățămîntul mediu din U.R.S.S.

Sub egida Academiei de Științe Pedagogice și a filialei din Siberia a Academiei de științe ale URSS, a fost propusă, încă din 1985, o programă pentru disciplina „Bazele informaticii și tehnicii de calcul”, destinată elevilor din clasele a IX-a și a X-a ale învățămîntului liceal din URSS.

Avînd în vedere seriozitatea cu care au fost tratate problemele, în cele ce urmează se prezintă o serie de idei și extrase din această programă.

De la început se apreciază că nivelul și calitatea pregătirii tineretului în domeniul tehnicii de calcul influențează direct potențialul științific-tehnic și productiv al unei țări.

La baza acestei discipline se află trei concepte fundamentale ale științei moderne: *informația, algoritmul și calculatorul*.

Prelucrarea automată a informației cu ajutorul calculatorului presupune următoarele etape:

- *algoritmizarea*: elaborarea algoritmului pentru soluționarea problemei date, în cadrul căreia intră studiul obiectului în cauză, descrierea obiectului din punctul de vedere al principalelor legături și dependențe între caracteristicile și proprietățile sale relevante; *modelarea matematică* — descrierea problemei într-un limbaj matematic formal; elaborarea metodei de studiu a modelului matematic și descrierea acestuia într-un limbaj algoritmic;
- *programarea*: traducerea algoritmului obținut într-un limbaj de programare, elaborarea programului pentru calculator;
- *soluționarea problemei cu ajutorul calculatorului*: execuția programelor pe calculator, folosind programele de sistem existente în calculator și obținerea rezultatelor sub o formă comodă pentru utilizare.

Scopul disciplinei este acela de a asigura formarea unor reprezentări privind regulile și metodele necesare soluționării problemelor cu calculatorul, de căpătare a deprinderilor necesare pentru utilizarea calculatoarelor personale, de înțelegere a rolului calculatorului în diferite activități economico-sociale și a perspectivelor de dezvoltare a tehnicii de calcul.



Informatica și calculatorul asigură baza necesară pentru studiul la un nivel calitativ superior a numeroase discipline din programa de învățământ. Astfel, folosirea calculatorului în predarea unor discipline permite: înțuirea mai profundă a unor noțiuni, modelarea unor obiecte și procese complexe, greu de urmărit și înțeles în absența calculatorului, sporirea gradului de participare a elevului la lecție, stimularea lucrului individual.

*Printre obiectivele disciplinei de informatică se evidențiază:*

- sistematizarea aspectelor algoritmice ale algebrei predate până în clasa a VIII-a inclusiv;
- însușirea noțiunilor de bază privind algoritmizarea;
- formarea reprezentărilor referitoare la posibilitățile de automatizare a execuției algoritmului;
- consolidarea laturilor practică și tehnică ale algoritmizării, legate de implementarea concretă a algoritmilor pe microcalculatoare, pentru diverse probleme date;
- cunoașterea bazelor tehnicii moderne de calcul, pornind de la studiul principiilor de operare ale microcalculatoarelor;
- însușirea noțiunilor fundamentale și a metodelor de elaborare a programelor într-un limbaj de programare;
- formarea reprezentărilor privind etapele de soluționare a problemelor cu ajutorul calculatorului;
- cunoașterea principalelor domenii de utilizare a tehnicii de calcul și a rolului ei în dezvoltarea societății.

*Informatica și tehnica de calcul se studiază pe parcursul a doi ani, în clasele a IX-a și a X-a.*

În clasa a IX-a sînt prevăzute 34 ore de curs, fără laborator, iar în clasa a X-a 68 — de ore, dintre care 34 ore de curs și 34 ore de laborator.

Pentru predarea acestei discipline se recomandă folosirea unor mijloace audio-vizuale: filme, casete-video, planșe, animație pe calculator, pachete de programe destinate procesului de învățământ.

### **Conținutul programei la disciplina: "Bazele informaticii și tehnicii de calcul"**

Acest curs trebuie să lărgască în mod substanțial aparatul/instrumentarul matematic de care dispune elevul, în sensul universalității acestuia în raport cu domeniile de utilizare, asigurînd tot odată specificul formulării și rezolvării problemelor cu calculatorul.

Acest aparat-instrumentar matematic cuprinde noi elemente teoretice:

- metode de reprezentare a principalelor caracteristici ale obiectelor, în forma necesară studierii modelelor matematice corespunzătoare cu ajutorul calculatorului;
- tipuri de procese algoritmice;
- elementele și sintaxa limbajului algoritmic, cu orientare spre limbajul natural;
- un limbaj de programare orientat spre un calculator și pachetele necesare de programe de sistem și aplicații;
- noi deprinderi practice, care permit elevului realizarea etapelor de algoritmizare, programare și soluționare a problemelor cu calculatorul.

*Clasa a IX-a (34 ore de curs).*

*Tema 1. Introducere (2 ore).*

Informatica. Noțiunile de informație și prelucrare a informației. Informatica și tehnica de calcul. Noțiunile de bază privind calculatorul. Schema bloc a calculatorului, unitățile de bază, funcțiunile și interacțiunea lor în procesul de operare.

Ideile de bază: informatica — știința despre metodele și mijloacele de soluționare a problemelor cu calculatorul.

Noțiunile de bază: informatica, tehnica de calcul.

Elevul trebuie să capete reprezentări privind: informatica — o nouă știință, interacțiunea dintre informatica și tehnica de calcul, calculatorul ca mijloc pentru prelucrarea automată a informației.



## *Tema 2. Algoritmi, limbaj algoritmic (6 ore).*

Noțiunea de algoritm. Exemple de algoritmi. Proprietățile algoritmilor. Mijloace de execuție a algoritmilor: operator uman, robot, calculator. Posibilitățile de descriere a algoritmilor.

Limbajul algoritmic ca mijloc de exprimare și scriere a algoritmilor. Comenzi simple și compuse. Condiții, comenzi de repetare și ramificare. Algoritmi ajutători, utilizarea lor. Exemple de scriere a algoritmilor în limbajul algoritmic. Posibilități de automatizare a execuției algoritmilor.

*Ideile de bază:* limbajul algoritmic ca formă unică, standard, pentru scrierea algoritmilor; posibilități de automatizare a execuției algoritmilor.

*Noțiunile de bază:* algoritmi, tipuri de algoritmi (liniari, cu ramificații, ciclici). Mijloace de descriere a algoritmilor.

*Elevul trebuie să cunoască:*

- esența algoritmilor, proprietățile lor, posibilitățile de execuție;
- modalitățile de bază privind descrierea algoritmilor (limbaj natural, tabele, scheme bloc);
- tipuri fundamentale de algoritmi (liniari, cu ramificații, cu cicluri);
- regulile de scriere a algoritmilor în limbaj algoritmic.

*Elevul trebuie să fie capabil:*

- să utilizeze diverse forme de scriere a algoritmilor și să treacă de la o formă de descriere, la alta formă.

## *Tema 3. Algoritmi de operare cu mărimi. (10 ore).*

Mărimi. Numele și valoarea mărimii. Mărimi variabile și constante. Noțiuni despre algoritmi care operează cu mărimi. Argumentele și rezultatele algoritmilor. Atribuirea de valori unei mărimi. Relațiile între mărimi, proprietățile mărimilor văzute în calitate de condiții simple. Condiții compuse. Tabele cu mărimi. Reguli de realizare a algoritmilor.

*Ideile de bază:* mărimile ca argumente și rezultate ale algoritmilor, care operează cu mărimi.

*Noțiunile de bază:* mărimea (nume și valoare), tipuri de mărimi; argumentele și rezultatele algoritmilor care operează cu mărimi;

*Elevul trebuie să cunoască:*

- tipurile de mărimi utilizate în prelucrarea informației (reprezentări: numerice, tabulare, literale, grafice) și semnificația operației de atribuire de valori unei mărimi;
- particularitățile algoritmilor care operează cu mărimi;
- scopul și regulile de scriere ale condițiilor de ramificare și repetare ale comenzilor algoritmilor.

*Elevul trebuie să fie capabil:*

- să determine tipul mărimii și să-l reprezinte în limbaj algoritmic;
- să folosească condițiile simple și compuse în construcția algoritmilor.

## *Tema 4. Construcția algoritmilor pentru soluționarea problemelor (16 ore).*

Etapele care se parcurg în soluționarea problemelor. Formularea problemei. Definirea argumentelor și a rezultatelor. Construcția modelului matematic. Alegerea algoritmilor ajutători. Aducerea expresiilor la o formă comodă pentru calcul. Calculul pe etape al expresiilor complexe. Microcalculatorul ca mijloc de calcul.

Construcția algoritmilor și soluționarea unor probleme din disciplinele de matematică, fizică și chimie.

*Ideile de bază:* tehnologia soluționării problemelor folosind calculatorul.

*Noțiunile de bază:* etapele soluționării problemelor folosind calculatorul.

*Elevul trebuie să cunoască:*

- etapele principale ale procesului de soluționare a problemelor cu ajutorul calculatorului;
- destinația și structura bibliotecii de algoritmi;
- posibilitățile calculatorului ca mijloc de calcul folosit de utilizator;



*Eleul trebuie să fie capabil:*

- să evidențieze etapele soluționării problemelor;
- să folosească algoritmi de bază din biblioteca pentru rezolvarea problemelor;
- să alcătuiască și să execute algoritmi pentru rezolvarea unor probleme de la disciplinele: matematică, fizică, chimie etc.

*Clasa a-X-a (34 ore curs).*

*Tema 5. Unitățile de bază și principiile de operare ale calculatorului. (12 ore).*

Informația binară, reprezentarea numerelor și textelor în calculator. Memoria calculatorului. Celula de memorie și adresa ei. Memoria internă (cu conținut permanent și variabil — memoria principală) și memoria externă, funcțiile lor. Resursele funcționale ale calculatorului: registre, sumator, unitatea aritmetică. Principiile de comandă ale calculatorului: instrucțiunea și structura ei, unitatea de comandă, execuția automată a programului. Introducerea și extragerea informației din calculator: tastatura, ecranul, banda magnetică, discul magnetic etc.

Generațiile de calculatoare. Elemente privind microprocesorul și circuitele integrate, tehnologii de fabricație.

*Ideile de bază:* posibilitatea execuției automate a programului într-un calculator.

*Noțiunile de bază:* unitățile principale ale unui calculator, destinația lor funcțională și principiile de operare.

*Eleul trebuie să cunoască:*

- particularitățile și avantajele formei binare de reprezentare a informației;
- tipurile de memorii folosite în calculatoare, destinația și principalele lor caracteristici;
- principiul comenzii după program;
- tipuri de echipamente de intrare / ieșire și destinația lor;
- elemente privind generațiile de calculatoare, microprocesoarele ca bază a tehnicii moderne de calcul.

*Tema 6. Programare. (16 ore).*

Program-algoritm, execuție pe calculator. Particularitățile calculatorului în calitate de executant al algoritmului. Comparatie între limbajul algoritmic și limbajul de programare. Reprezentarea datelor. Variabile, Instrucțiunile de bază: intrare, ieșire, atribuire, control al execuției programului. Subprograme și funcțiile standard ale limbajului. Întocmirea programelor în limbaj de programare. Elemente privind pachetele de programe de sistem și aplicații ale calculatorului.

*Ideile de bază:* program-algoritm, execuție pe calculator.

*Noțiunile de bază:* limbajul de programare, folosirea lui pentru scrierea algoritmilor, pachetele de programe ale calculatorului.

*Eleul trebuie să cunoască:*

- posibilitățile și particularitățile calculatorului în calitate de executant al algoritmului;
- alfabetul limbajului de programare;
- tipurile de date ale limbajului și destinația lor;
- instrucțiunile de bază ale limbajului;
- destinația subprogramelor și funcțiilor standard;
- elemente privind pachetele de programe de bază și aplicații ale calculatorului și folosirea lor pentru soluționarea problemelor.

*Eleul trebuie să fie capabil:*

- să scrie/reprezinte datele în limbajul de programare;
- să alcătuiască programe simple în limbajul de programare, folosind instrucțiunile de bază: intrare/ieșire, atribuire, control al execuției programului
- să folosească funcțiile standard și subprogramele standard;
- să utilizeze pachetele de programe aplicative.

*Tema 7. Rolul calculatorului în societatea modernă. Perspectivile dezvoltării tehnicii de calcul. (2 ore).*

Aplicații ale calculatoarelor: calcule ingineresti și economice, modelare matematică sisteme de reglare automată, sisteme de informare-documentare, automatizarea locurilor de



muncă etc. Rolul calculatorului în schimbarea caracterului și ridicarea productivității muncii. Perspectivile dotării economiei naționale cu tehnică de calcul.

*Ideile de bază:* caracterul universal al calculatorului, utilizarea lui în diferitele domenii ale activității sociale, rolul calculatorului în societatea contemporană, schimbarea caracterului muncii și ridicarea productivității acesteia.

*Noțiunile de bază:* domeniile în care se utilizează calculatorul, perspectivele de dezvoltare a tehnicii de calcul.

*Elevul trebuie să cunoască:*

— principalele domenii de utilizare a calculatoarelor și rolul tehnicii de calcul în societatea contemporană.

*Tema 8. Vizită la un centru de calcul. (4 ore)*

\* \* \*

În scopul dezvoltării la elevi a interesului pentru studiul informaticii și tehnicii de calcul, cit și a ridicării nivelului lor de cunoștințe, se recomandă intensificarea activităților cu caracter facultativ, și a celor în afara programei de învățământ: cercuri de elevi, participare la concursuri și olimpiade de informatică.

## 16.4. Aplicații informatice educaționale în diferite țări

### 16.4.1. Informatica în școală. Experiența franceză

(Jean Yves Chateau — Chargé de Mission Ministère de l'Education Nationale, de la Jeunesse et des Sports, Direcția școlilor, Paris, France) (30 martie 1989)

Poarte dificile probleme, datorită deciziei din 1985, de generalizare imediată a informaticii.

*Echipamentele* din 1985 sînt insuficiente pentru o utilizare semnificativă și pentru toți profesorii, au o putere corespunzătoare unor programe de învățământ asistat de calculator și de jocuri puțin ambițioase care păreau în acel timp cel mai bun aport la informatica școlii elementare, dar care apoi sînt depășite de programe generalizabile care sînt adevărate unelte de învățământ (prelucrare de texte, tabele electronice, simulări, ilustrații). Programele existente care nu putem pretinde că au, prin ele, însele, virtuți pedagogice, dar pot servi într-un mod adecvat și eficace interesului didactic al disciplinelor, nu sînt prea numeroase, mai ales că nu sînt într-o formă adaptată nevoilor școlare și capacităților copiilor.

*Profesorii de școli elementare* (circa 170 000) din care imensa majoritate nu a primit o formație inițială sistematică în domeniu, de altfel greu de conceput — la acest mare număr —, nepărînd satisfăcătoare nici o simplă lecție de conducere și pilotare a calculatorului nici cunoașterea unui program generalizabil reputat (oricum, mai rațională).

*Evaluarea programelor* este încă neprecisă, iar formarea profesorilor necesită timp și elemente solide din domeniile științific, epistemologic, didactice, pedagogice și informatice, foarte eterogene.

*Măsurile luate* nu elimină aceste probleme dar pot fragmenta dificultățile; în ultimii doi ani a fost programată o cultură informatică de nivel mediu și sînt la dispoziție (la cerere) mijloace informatice pentru învățământ. Este foarte urgentă formarea a 50 000 de profesori de curs mediu, care vor trebui să dezvăluie elevilor realitatea socială, științifică și tehnologică a informaticii în lume și în mediul înconjurător, la școală și în afara ei, la un nivel de înțelegere cît mai mare posibil. Cea mai mare masă a învățătorilor trebuie să utilizeze un program generalizabil sau altul, oportun în cadrul lor de învățământ, sarcină mult mai delicată ce trebuie aplicată progresiv. Calitatea programelor pentru învățământul elementar este în creștere lentă dar sigură.

Școlile franceze dispun azi de 105 000 posturi de lucru, la cca 40 000 școli și 170 000 clase, adică 2,5 posturi / școală, 1,6 clase / post și 35 elevi pe post. Aparent, Franța se află printre primele două sau trei țări din acest punct de vedere (dacă nu se iau în considerare



controversele între academii și departamente). Pentru clasele de curs mediu media este de 13, satisfăcătoare; pentru acest curs mediu există un număr convenabil de posturi de lucru, activitatea informatică fiind în aproape 90% din clase, în ce privește utilizarea programelor și mai mult de 70% din învățători ce se ocupă de informatică.

Există un *dispozitiv național* care se ocupă în toate departamentele de informatică, dând sfaturi pentru achiziția programelor. Alături de formarea inițială de minim 70 ore există și alte acțiuni: stagii de una sau mai multe săptămâni în timpul de lucru, ateliere — cu formare „à la carte”, ședințe de susținere ținute de învățători specialiști.

Comunele sînt, exceptînd echipamentul finanțat de stat în 1985, responsabile de echipamente și programe; în 1987 a început o politică de legătură sistematică: stagiile de formare continue din Școala Normală implică împrumuturi de programe la sfîrșitul stagiului, fapt ce permite achiziții în cunoștință de cauză.

*Practicile pedagogice legate de informatică.* Evaluarea cu calitate pedagogică, la 4 ani de la generalizare a informaticii e nesigură; o evaluare va face Inspekția generală; oricum, practica pedagogică este de calitate inegală; nu se știe clar cît timp consacră elevii informaticii; de asemenea, din punct de vedere pedagogic, nu e relevant, cînd se adună programe de complexități diferite și se face o medie (care e greșită). Eforturile de cuantificare premature riscă să golească de semnificația calitativă rezultatele.

Pentru a evalua din punct de vedere calitativ informatica în școală este satisfăcător să apreciem programele cele mai utilizate în clase, care ne permit să știm ce tipuri de activități se derulează în clase (LOGO, prelucrări de texte etc.). O anchetă din 1988 a arătat că, drept suport de instruire, două treimi din clasele elementare utilizează programe generalizabile. O anchetă a 4 828 învățători (1 055 școli) au menționat 450 produse-program diferite: două programe generalizabile sînt la mare distanță de celelalte (un sfert din clasele ce utilizează informatica folosesc prelucrarea textelor iar 50% din ele folosesc LOGO) ca excepții; se utilizează produse educaționale asistate de calculator, programe de matematică, logică, combinatorică, limbă (scriere, lectură, gramatică).

În cursurile preparatorii mai puțin de 50% din clase utilizează produse program, în cursurile elementare — mai mult de 50%. În cursurile medii — aproape de 90%. Învățătorii celor mai tineri elevi nu sînt prea convinși de aplicațiile informatice ce se propun. De la 9—10 ani utilizarea informaticii le pare posibilă și utilă, fapt concurent cu marile lucrări clasice de psihologie genetică (Piaget, Vignon); informatica produce un mixaj original de concret și abstract, de trecere de la stadiul operațiilor concrete la cel de operații formale (de aceea Franța a dat importanță în 1985 cursurilor medii). Acest fapt al unui studiu, chiar rudimentar, este un factor de favorizare a produselor programe-educaționale.

## 16.4.2. Informatica în învățămîntul general. Planul japonez

(Harno Nishinosono, Perspectives nr. 4. 1987)

Japonia a avut o *întîrire mare* în introducerea calculatoarelor în învățămînt (în 1983 calculatoarele erau prezente în 0,1% școli primare, 1,8% gimnazii, 45,6 secundare, iar în 1985 datele erau 2,1%; 13,8%; 80,6%). *O excepția învățămîntului secundar* (liceal) unde este un pionier pentru formarea tehnicienilor și informaticienilor. Rapoartele guvernamentale din 1985, 86, 87 au dat impulsul informaticii educaționale pe trei niveluri (primar, secundar — ciclul I, secundar — ciclul II) și în 3 moduri (calculatorul ca unealtă pedagogică, ca materie de învățămînt, ca mijloc — planuri cursuri, gestiune) pentru a atinge scopuri ca: realizarea obiectivelor fundamentale ale învățămîntului, dezvoltarea de noi calități la elevi, adaptarea instruirii la nivelul elevului, utilizarea de medii pentru animarea vieții școlare, crearea de condiții indispensabile eficacității învățămîntului.

*Pregătirea instructorilor* este esențială; ea se realizează în facultățile de științe ale universităților, în colegii, în institute de pedagogie; programele acestora — în cooperare cu industria și specialiștii informaticieni vor suferi modernizări, utile pentru viitor.

*În prezent* se propagă pregătirea instructorilor în centrele locale (48 prefecturi) de prelucrare a datelor și institutele locale de cercetări pedagogice prin cursuri, în general scurte și desfășurate mai ales în vacanțe.

Astfel, un curs elementar în 3 zile, un curs mediu în 5 zile, cu curs superior în 8 zile. Aceste cursuri ar putea avea următorul conținut:



### *Cursul elementar*

*Teorie* (3 ore) — rolul calculatorului în societate; calculatorul personal, hardware și software; programe și date; utilizare în învățământ situație actuală.

*Manipulare* (3 ore) — organizarea aparatelor și manipularea tastaturii; utilizarea memoriilor.

*Tehnici* (6 ore) — analiza sarcinilor și organigrama; introducerea și executarea unui program elementar; salvarea unui program sau a datelor prin trecerea în memorie.

*Aplicații* (3 ore) — (trunchi comun = tr. c) posibilități de utilizare a calculatorului în învățământ; — în învățământul școlar; — în educația socială.

### *Cursul mediu*

*Teorie* (3 ore) — tr.c., aplicații ale c. în situații reale din învățământ; — tr.c., tipuri de software și de limbaje de programare; — inv. școlar, utilizarea c.; — educație socială, utilizare c.

*Manipulare* (3 ore) — configurația și manipularea diferitelor periferice de intrare și ieșire.

*Tehnici* (12 ore) — tr.c., tehnici de analiză a sarcinilor și a programării; — tr.c. încercări de diferite limbaje de programare; — inv.școlar, elaborarea unui program pentru o problemă precisă; educație socială, elaborarea unui program pentru o problemă precisă.

*Aplicații* (12 ore) — tr.c., evaluarea și gestionarea programelor; tr.c., tipuri de informații utilizabile în învățământ și în practică, structura datelor; — tr.c., conținutul și metodele unei inițieri în informatică; inv. școlar, utilizarea calculatorului în clasă; — inv. școlar, prelucrarea datelor în școală; inv. școlar, studiu de caz; educație socială, utilizarea calculatorului; — educație socială, prelucrarea informației în instituții de educație socială; educație socială, studiu de caz.

### *Cursul superior*

*Teorie* (6 ore) — tr.c., principiul și realitățile utilizării calculatorului în învățământ; tr.c., procedură și tehnici de elaborare a sistemelor de utilizare a calculatoarelor; — inv. școlar, elaborarea de sisteme pentru utilizarea calculatorului în învățământul școlar; educație socială, elaborare de sisteme pentru educație socială.

*Manipulare* (3 ore) — utilizarea echipamentelor pentru elaborarea sistemelor.

*Tehnici* (21 ore) — tr.c., elaborarea de sisteme, studiu de caz; — inv. școlar, conceperea unui sistem specific pentru utilizarea calculatorului în învățământul școlar; — ed. socială, concepția unui sistem specific pentru utilizarea calculatorului în educația socială.

*Aplicații* (18 ore) — tr.c., funcția socială și educativă a mijloacelor de informație și de comunicație — tr.c., tehnici de evaluare, gestiune și de manipulare a unui sistem informațional; — tr.c., metode pedagogice de formare a utilizării calculatorului; — învățământul școlar, conceperea conținuturilor învățământului cu microcalculatoare; — învățământul școlar, compunerea sistemelor de prelucrare a informației cu finalitate educativă și construcția unei baze de date; — inv. școlar, planificarea formării și utilizării calculatoarelor în inv. școlar; — ed. socială, dezvoltarea activităților sociale de educație și utilizarea calculatorului; — educație socială, conceperea sistemelor de prelucrare a informației cu finalitate educativă și construcția unei baze de date; educație socială, planificarea formării / utilizării calculatorului în educația socială.

## 16.5. Concursuri de informatică în țară (probleme propuse și rezolvate)

### 16.5.1. Concursul național de informatică 14–22 iulie 1987

#### PROBLEME PROPUSE

##### *Clasa a V-a*

Se dă un șir de numere naturale. Găsiți valoarea maximă și valoarea minimă, iar cu ajutorul lor desenați un dreptunghi ale cărui laturi sînt aceste valori. În interiorul dreptunghiului scrieți aria sa.

##### *Clasa a VI-a*



Să se genereze două șiruri A și B de numere naturale. Elementele primului șir sînt cuprinse între 0 și 99, iar elementele celui de al doilea șir sînt cuprinse între 1 și 999. Cele două șiruri au același număr de elemente care este specificat de la tastatură. Utilizînd elementele șirurilor A și B să se genereze un șir C după următoarea regulă:

- $C_i = A_i - B_i$  dacă  $A_i$  este par și  $B_i$  este impar;
- $C_i = A_i + B_i$  dacă  $A_i$  este impar și  $B_i$  este par;
- $C_i = A_i * B_i$  dacă  $A_i$  și  $B_i$  sînt ambele fie pare, fie impare.

Să se afișeze, cîte patru elemente pe un rînd, elementele șirului B apoi elementele șirului C.

#### Clasa a VII-a

Se citesc maxim 15 perechi de numere naturale ( $X_i$ ,  $Y_i$ ) ce reprezintă coordonatele unor puncte.

Să se scrie un program care ordonează crescător aceste perechi după valorile  $X_i$  și unește prin segmente de dreaptă aceste coordonate.

Avînd în vedere că aceste perechi pot avea valori numerice oarecare se cere o normalizare a acestora pentru a putea fi reprezentate optim pe ecran într-un dreptunghi cu laturile de 200 respectiv 150 puncte.

#### Clasa a VIII-a

Se citesc datele despre un grup de maxim 20 de elevi. Fiecare elev este specificat prin: Nume, Prenume, Nota Mat, Nota Rom, Nota Piz. Se cere să se elaboreze un program care afișează grupul de elevi, cu toate datele despre ei, la cerere astfel:

- a) în ordine alfabetică;
- b) în ordine descrescătoare a mediilor celor trei note;
- c) specificîndu-se un nume de elev să se reprezinte grafic notele și media acestuia sub forma unui dreptunghi cu baza aceeași și înălțimea proporțională cu nota, respectiv media. Fiecare dreptunghi va fi colorat cu altă culoare.

## REZOLVĂRI

### Soluția problemei de clasa a V-a 1987

```

10 REM citește șirul de numere
20 INPUT "Nr. de elemente :";n
30 DIM a(n)
40 FOR i=1 TO n
50   PRINT "a(";i;")=";
60   INPUT "valoare element = "; a(i)
70   PRINT a(i)
80 NEXT i
90 REM calculează minim și maxim
100 LET min=a(i):
   LET max=a(i)
110 FOR i=1 TO n
120   IF min > a(i) THEN LET min=a(i)
130   IF max < a(i) THEN LET max=a(i)
140 NEXT i
150 REM calculează aria
160 LET aria = max*min
170 CLS
180 PRINT "valoare maximă = "; max
190 PRINT "valoare minimă = "; min
200 REM desenează dreptunghi
20 IF max > 150 THEN GO TO 270
220 IF max < 72 THEN GO TO 300
230 PLOT 0,0
240 DRAW max, 0
   DRAW 0, min
   DRAW -max, 0
   DRAW 0, -min
250 PRINT AT 21-min/16, max/16-6; "aria="; aria
260 STOP

```



```

270 INPUT "cu ce scară doriți să micșorați? ";s
280 LET max=max/s:
    LET min=min/s
290 GO TO 210
300 INPUT "cu ce scară doriți să măriți? ";s
310 LET max=max*s:
    LET min=min*s
320 GO TO 210

```

#### Soluția problemei de la clasa a VI-a 1987

```

10 REM programul generează un șir de numere naturale pe bază a două șiruri
20 INPUT "Nr. de elemente ale șirurilor="; n
30 DIM a(n):
    DIM b(n):
    DIM c(n):
    DIM e(n):
40 REM generare elemente a(i) și b(i)
50 FOR i=1 TO n
60     LET a(i)=INT (100•RND):
        LET b(i)=1+INT (999•RND)
70 NEXT i
80 REM generare elemente șir c
90 FOR i=1 TO n
100     LET ra=a(i)-2•INT (a(i)/2):
        LET rb=b(i)-2•INT (b(i)/2)
110     IF ra=0 AND rb=1 THEN LET c(i)=a(i)-b(i)
120     IF ra=1 AND rb=0 THEN LET c(i)=a(i)+b(i)
130     IF ra=rb THEN LET c(i)=a(i)•b(i)
140 NEXT i
150 PRINT
160 PRINT "Elementele șirului a : "
170 FOR i=1 TO n:
    LET e(i)=a(i):
    NEXT i
180 GO SUB 290
190 PRINT
200 PRINT "Elementele șirului b : "
210 FOR i=1 TO n:
    LET e(i)=b(i):
    NEXT i
220 GO SUB 290
230 PRINT
240 PRINT "Elementele șirului c : "
250 FOR i=1 TO n:
    LET e(i)=c(i):
    NEXT i
260 GO SUB 290
270 PRINT
280 STOP
290 REM subrutina de tipărire șir
300 REM intrări — e(i) elementele șirului
310 REM ieșiri — tipărește elementele șirului
320 LET x=0
330 FOR i=1 TO n
340     PRINT TAB 7•x; e(i):
350     LET x=x+1
360     IF x=4 THEN PRINT:
        LET x=0
370 NEXT i
380 RETURN

```

Paragraful 16.5  
continuă la anexe (vol. 2)



## 16.6. Teme rezolvate pentru cluburile de informatică

În acest subcapitol se propun și se dau soluții pentru o serie de teme de programe axate pe matematică, fizică, chimie, electronică și astronomie, ilustrând o varietate de obiective, ca de pildă: demonstrații și jocuri educative, testarea cunoștințelor, asistarea profesorului și elevului în procesul de predare, respectiv de învățare, rezolvarea cu calculatorul a unor clase de probleme și simularea unor situații reale, explorate în mod variat.

Temele implică elaborarea unor programe noi, complexe, sub îndrumarea profesorului de informatică.

Pentru temele nr. 9 și 11, de trigonometrie și fizică, sînt prezentate în detaliu programele TRIGRAF și ACCO, oferite și pe minicasetă, ca soluții posibile și exemple de lucru. De asemeni — pentru temele 3, 4, 8, 13, 18, 20 ș.a.; alte programe apropiate temelor în cap. 17 și 18 (71 programe).

Temele au fost schițate pe baza experienței acumulate în licee, gimnazii și școli primare din București, a experienței acumulate la Institutul Politehnic precum și la Institutul de tehnică de calcul și I.C.I. din București, ținîndu-se seama de cîteva recomandări și experiențe încurajatoare prezentate în literatura de specialitate. (v. și anexa)

**Pentru indicații și soluții a se vedea și anexa de la sfîrșitul volumului.**

(Temele 1, 2, 3, 5, 10, 13)

### 16.6.1. Programe instructiv-educative

*Tema nr. 1:*

Programe demonstrative privind facilitățile grafice color ale calculatorului HC-85. Se vor trasa figuri plane și în spațiu, inclusiv modele amuzante, dar ilustrative, ca de pildă „pălăria de mexican” descrisă de o funcție mai complicată:

$$z = (\cos r)e^{-r/3}$$

unde

$$r = x^2 + y^2$$

Acest program pentru trasarea funcției  $z$  în trei coordonate (3D) are 10 linii, dar durează circa 25 minute.

*Tema nr. 2:*

Programe de șah cuprinzînd problema trasării unei table de șah cu figurarea pieselor și o rezolvare de problemă de șah, ca de pildă problema celor 8 dame.

Această problemă implică amplasarea pe tabla de șah (deci într-o matrice de  $8 \times 8$ ) a 8 dame care să nu se atace, fiind necesar să se calculeze și afișeze cele 92 de soluții ale problemei, de fapt 23 de soluții distincte, reluate prin simetrie.

### 16.6.2. Testarea cunoștințelor

*Tema nr. 3:*

Programul ARP de testare a cunoștințelor despre suprafețe plane (pătrat, dreptunghi, cerc și elipsă). Se trasează un desen complex cu mai multe



figuri geometrice înscrise succesiv: elipsa, cercul de diametru mic, cercul de diametru mare, elipsa înscrisă într-un dreptunghi și cercul mare înscris într-un pătrat. Pe desen se dau diametrul mare (2 a) și diametrul mic (2 b). Se va da un exemplu de testare a răspunsului privind o formulă de arie plană și apoi se formulează minim 5 întrebări de testare a cunoștințelor despre arii, fiind obligatoriu ca răspunsul la fiecare întrebare să fie testat într-un timp dat (de pildă, 60 de secunde).

O întrebare interesantă poate cere să se precizeze dacă aria elipsei este mai mare, mai mică, sau egală cu semisuma ariilor cercurilor de diametru mic și de diametru mare (corespunzătoare semiaxelor elipsei).

Pentru elevii mai mici, se poate indica opțional aria elipsei, sugerându-se o rezolvare algebrică.

Se contorizează răspunsurile corecte și în final, se acordă ca premiu câte un desen din ce în ce mai atractiv, în funcție de numărul de răspunsuri corecte.

#### *Tema nr. 4:*

Programe de testare a cunoștințelor despre operații aritmetice, formule algebrice, de geometrie și trigonometrie, corelate cu diverse niveluri de pregătire. Se va cere încadrarea răspunsurilor în intervale date de timp (modificabile în funcție de etapa de pregătire). În cazul unor răspunsuri eronate pentru cunoștințe obligatorii, se vor oferi informații ajutătoare. Se contorizează răspunsurile corecte și se afișează punctajul obținut.

Se pot elabora programe de testare a cunoștințelor pentru fizică, chimie, limba română, limbile străine, geografie, istorie, muzică, desen, practic pentru orice disciplină de învățământ sau domeniu, utilizând atractiv și eficient facilitățile grafice și sonore ale calculatorului.

### **16.6.3. Asistarea profesorului în procesul de predare și a elevului în procesul de învățare**

#### *Matematica*

#### *Tema nr. 5:*

Programul GRAF 2, pentru cunoașterea funcției de gradul al II-lea, rezolvarea ecuației, trasarea graficului funcției și rezolvarea grafică a unor sisteme de ecuații de tipul

$$\begin{cases} y = ax^2 + bx + c \\ y = mx + n \end{cases}$$

unde  $a, b, c$  sînt coeficienți reali, iar ecuația liniară are parametrii  $m$  și  $n$  reali.

Se vor analiza cazurile în care apar module ale funcțiilor respective (la una din funcții sau la ambele).

Calculatorul este util în următoarele faze ale lecției sau lecțiilor:

- prezentarea sugestivă a semnificației coeficienților funcției de gradul al II-lea;
- ilustrarea injectivității și surjectivității unor funcții;

### **VII. C.P. ÎN ÎNVĂȚĂMÎNT ȘI EDUCAȚIE**



— trasarea graficului plecând de la forma  $ax^2$ , prin translație verticală și apoi orizontală a parabolei, inclusiv calculul intersecțiilor cu axele și al coordonatelor vârfului parabolei, deci rezolvarea ecuației;

— rezolvarea grafică a sistemelor de mai sus, ilustrând cazurile de compatibilitate și numărul de soluții pentru sistemele compatibile.

O problemă interesantă de studiat ar fi trasarea opțională a graficului funcției de gradul II pe intervalele dominante, cu glisarea automată a originii axelor (efect de lupă) în funcție de mărimea parametrilor principali de poziționare, astfel încât zona cea mai mare a ecranului să fie alocată cadranelui sau cadranelor în care se trasează funcția.

#### *Tema nr. 6: (v prob. 17.21, 17.23)*

Programe privind calculul de arii și volume pentru corpuri de rotație.

Se generează pe ecran corpurile de rotație, facilitând elevilor rezolvarea problemelor respective prin identificarea rapidă a parametrilor și proprietăților corpurilor de rotație.

Se va propune elevilor rezolvarea unor probleme diverse pentru corpuri obținute prin rotirea unor figuri plane cum sînt: triunghiul isoscel în jurul înălțimii relative la bază; triunghiul dreptunghic în jurul catetelor; dreptunghiul în jurul mediatoarelor laturilor; trapezul isoscel în jurul axei de simetrie, al uneia din laturile egale sau în jurul unei baze; hexagonul regulat în jurul axelor de simetrie, etc.

#### *Tema nr. 7:*

Programul TREP pentru învățarea teoremei celor trei perpendiculare și a reciprocelor ei.

Pe ecran se generează o imagine a construcției geometrice în spațiu, apar datele ipotezei, iar demonstrației date pe ecran i se asociază efecte grafice și sonore pentru a ilustra perechile de drepte perpendiculare.

Se vor introduce caractere speciale pentru semnele:

$\in, \notin, \subset, \not\subset, \perp, \alpha, \leq, \exists, \cap, \neq$  și  $\emptyset$

Opțional, pentru verificarea cunoștințelor dobîndite, prin program se va cere elevului să tasteze concluzia teoremei, care apare ca firească în finalul demonstrației teoremei directe, sau al reciprocelor ei.

#### *Tema nr. 8:*

Program pentru ilustrarea graficelor funcțiilor trigonometrice  $\sin x$  și  $\cos x$ , construite prin puncte, utilizînd cercul trigonometric pe care se deplasează un punct, în timp ce în altă zonă a ecranului, se completează un tabel cu perechile de valori ale argumentului și funcției, pentru multiplii lui  $\frac{\pi}{4}$  și  $\frac{\pi}{6}$ , respectiv se figurează punctele principale ale graficului care în final se unesc. Opțional, tabelul se poate șterge, rămînînd numai graficul funcției și cercul trigonometric.

Pe baza rezultatelor grafice se vor introduce și discuta intuitiv unele proprietăți ale funcțiilor  $\sin x$  și  $\cos x$  ca semnul, imparitatea/paritatea, monotonia, injectivitatea, surjectivitatea.

Se va elabora programul și pentru funcțiile  $\operatorname{tg} x$  și  $\operatorname{ctg} x$ .

(v. și pe casetă)



### *Tema nr. 9 și un exemplu de rezolvare:*

Programul TRIGRAF pentru studiul, trasarea și mixarea funcțiilor trigonometrice  $\sin(nx)$ ,  $\cos(nx)$  și  $\operatorname{tg}(nx)$ .

Mixarea graficelor pentru funcții diferite de același argument, sau pentru aceeași funcție dar cu argumente diferite, permite studiul comparativ al funcțiilor respective și ilustrarea grafică a unor proprietăți.

În subcapitolul 18.41 și pe casetă se prezintă programul TRIGRAF pentru studiul și trasarea funcțiilor trigonometrice  $\sin$ ,  $\cos$  și  $\operatorname{tg}$ , în cadrul unei lecții de matematică, respectiv programul scris în BASIC — HC 85 și comentariile aferente.

Programul TRIGRAF reprezintă un exemplu de rezolvare a temei nr. 9, dar elevilor care au depășit faza inițială de introducere în limbajul BASIC — HC 85, li se poate cere să pună la punct o versiune proprie a programului sau cel puțin să modifice programul TRIGRAF pentru a include și funcția  $\operatorname{ctg}(nx)$ .

### *Tema nr. 10 și o sugestie de rezolvare: (v. și ANEXE):*

Program pentru trasarea graficului oricărei funcții în intervale selectabile. Programul poate fi elaborat în mai multe etape, vizînd următoarele obiective:

1) realizarea analizei lexicale și sintactice a expresiei funcției în scopul determinării corectitudinii acesteia și al stabilirii atomilor lexicali (constante, identificatori, operații, funcții și delimitatori), cu afișarea tipului de eroare;

2) trasarea funcțiilor fără asimptote verticale, cu scalare automată. Pentru scalare, se va utiliza complet spațiul de afișare al ecranului, folosind atît intervalul dat  $[x_1, x_2]$ , cît și valorile extreme ale funcției (minim și maxim) pe intervalul respectiv;

3) determinarea punctelor în care funcția este nedefinită sau nedeterminată (avînd asimptote verticale);

4) introducerea posibilității de a mări diferite zone ale graficului (efect de lupă), cu marcarea punctelor de inflexiune, de extrem, de întoarcere și unghiulare. Selectarea zonei de mărit, se poate face în mod grafic;

5) asigurarea facilității de a selecta scalarea automată sau scara 1:1;

6) extinderea posibilității de studiere de la o funcție la o familie de funcții parametrizate, dînd valori particulare setului de parametri;

Tema avînd o complexitate apreciabilă, este propusă mai ales pentru elevii avansați în domeniul informaticii, elaborarea versiunii proprii a programului, fiind de dorit să se desfășoare sub îndrumarea instructorului sau profesorului de informatică și numai după consultarea temeinică a acestei lucrări și a altor cărți de specialitate.

Programul SUPERGRAF, beneficiînd de o interfață prietenoasă cu utilizatorul, poate fi folosit de o gamă largă de elevi pentru studiul graficelor funcțiilor, fiind de asemenea util ca instrument auxiliar al profesorului de matematică în procesul de predare a lecțiilor despre funcții. În figura alăturată se dă o imagine-ecran, avînd trasată funcția  $F(x) = 1 + x + 2x \ln \left| \frac{x}{1-x} \right|$  cu ajutorul programului SUPERGRAF.



Față de facilitățile acestui program, propunem să se studieze problema selectării la afișare a numărului de zecimale și rezolvarea ridicării la putere de tip  $a(x)^n$ , unde funcția  $a(x)$  ia și valori negative.

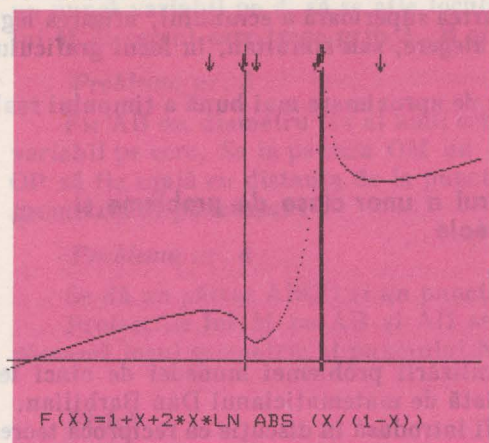


Fig. 16.1. Trasarea unei funcții cu programul SUPERGRAF

## Fizică

### Tema nr. 11 și un exemplu de rezolvare.

Programul educativ ACCO pentru studiul mișcării rectilinii uniform variate, facilitând înțelegerea noțiunii de accelerație și a condiției necesare pentru oprirea unui mobil ( $v=0$  și  $a=dv/dt=0$ ).

Problema constă în manevrarea de la tastatură a unui mobil, prin comenzi de accelerare/decelerare, până la oprirea acestuia într-un spațiu dat, marcat pe ecran de un steag și de un zid (figurat ca marginea din dreapta a ecranului), fără vreo oprire anterioară. În partea superioară a ecranului se generează graficul vitezei în funcție de spațiul parcurs, se afișează valoarea maximă admisă a vitezei (limitare ilustrată prin marginea de sus a ecranului) și valorile momentane ale celor trei parametri principali — timpul, viteza și accelerația, date la intervale de 0,3 s, cu o bună aproximare a timpului real (vezi figura de la subcapitolul 18.42).

Urmărind graficul vitezei și valorile parametrilor, se va constata că mobilul nu se oprește dacă viteza trece prin valoarea zero la un moment dat, iar accelerația este nenulă, fiind necesar ca viteza să ia valoarea zero în două momente de timp consecutive, deci accelerația să fie nulă.

Programul educațional de fizică ACCO oferă un exemplu de soluționare a temei definite mai sus.

În capitolul 18 se prezintă distinct utilizarea programului ACCO în cadrul unei lecții de fizică, respectiv programul ACCO elaborat în BASIC și comentariile aferente.

Pe baza acestei teme se va cere elevilor să elaboreze o versiune proprie a programului ACCO în BASIC sau LOGO. De asemenea, acest program poate fi dezvoltat în continuare la cercurile de fizică și informatică prin modificarea condițiilor inițiale ( $x_0$ ,  $v_0$ , deplasare cu frecare  $\mu$ , deplasare pe plan înclinat, distanța dintre steag și zid, etc.), redesenarea mobilului și introducerea unor efecte



sonore corespunzătoare accelerării/decelerării, atingerii liniei steagului și a zidului.

În altă versiune care se poate elabora sub îndrumarea instructorului de informatică, elevul poate obține în partea superioară a ecranului, afișarea legii vitezei și legii spațiului, separat, la alegere, sau simultan, în locul graficului vitezei în funcție de spațiu.

De asemenea, se vor studia căile de aproximare mai bună a timpului real.

#### 16.6.4. Rezolvarea cu calculatorul a unor clase de probleme și simularea unor situații reale

##### *Matematică*

##### *Tema nr. 12:*

Program pentru ilustrarea generalizării problemei monedei de cinci lei (problema lui Țițeica), generalizare dată de matematicianul Dan Barbilian.

Problema lui Țițeica, care poate fi introdusă în discuție ca reciproca teoremei lui Carnot, arată că dacă trei cercuri congruente avînd un punct comun  $H$  se mai intersectează două cîte două, respectiv în  $A$ ,  $B$  și  $C$ , atunci cele trei cercuri sînt congruente cu cercul circumscris triunghiului  $ABC$ .

Generalizarea constă în numărul arbitrar de cercuri congruente avînd un punct comun. Se ilustrează pe ecran cazurile a două cercuri congruente avînd un punct comun  $H$ , apoi trei, patru și cinci cercuri congruente cu un punct comun, fiecare caz fiind bazat pe cel precedent. Țițeica intuise faptul că această problemă se poate extinde la cazul a cinci piese. Experimentul conduce la concluzia că unui număr par de cercuri congruente avînd un punct comun, i se asociază un punct, iar unui număr impar de cercuri congruente avînd un punct comun, i se asociază un nou cerc de aceeași rază.

Elevii vor identifica astfel, o modalitate de obținere de rezultate noi în știință și anume calea:

intuiție  $\rightarrow$  verificare  $\rightarrow$  demonstrație.

Verificarea necesită tratarea unui mare număr de cazuri și calculatorul personal poate facilita confirmarea unei intuiții sau negarea ei și obținerea unor contraexemple.

##### *Tema nr. 13: (v. și ANEXE):*

Programe pentru asistarea însușirii noțiunilor despre cerc și aplicarea acestora în rezolvarea unor probleme de loc geometric.

Calculatorul este utilizat pentru ilustrarea noțiunilor de bază și pentru intuirea locului geometric cerut pe cale inductivă, eventual pentru verificarea corectitudinii locului geometric.

Se propune rezolvarea asistată de calculator a următoarelor probleme de loc geometric:

##### *Problema nr. 1*

Două cercuri se intersectează în  $A$ ,  $B$ . O secantă variabilă trecînd prin  $A$  taie cercurile a doua oară în  $M$ ,  $N$ . Să se afle locul geometric al mijlocului segmentului  $MN$ .



### Problema nr. 2

Se dau punctele distincte A, B și dreapta d perpendiculară pe AB. M fiind un punct variabil pe d, să se afle locul geometric al punctului diametral opus lui M în cercul care trece prin A, B și M.

### Problema nr. 3

Fie AB un diametru fix al unui cerc de centru O și rază R, iar M un punct variabil pe cerc. Se ia pe raza OM un punct P astfel ca mărimea segmentului OP să fie egală cu distanța de la punctul M la dreapta AB. Să se afle locul geometric al punctului P.

### Problema nr. 4

Se dă un pătrat ABCD și un punct M variabil pe diagonala BD.

Proiecțiile lui M pe AB și AD se notează cu E și, respectiv, F. Să se găsească locul geometric al punctului N de intersecție a dreptelor CF și DE.

## Fizică

### Tema nr. 14

Program pentru simularea experienței lui Millikan, de măsurare directă a sarcinilor electrice elementare — temă propusă pentru elevii avansați.

O serie de experiențe interesante de fizică sau chimie nu pot fi efectuate datorită condițiilor de desfășurare a acestora, complexității, cerințelor de precizie, costului ridicat, gradului de pericolozitate, condiții care adesea nu pot fi întrunite în laboratoarele școlare.

Un exemplu este experiența lui Millikan (descrișă în manualul de fizică pentru clasa a XII-a), privind determinarea directă a sarcinii electronului.

Deoarece realizarea acestui experiment, incluzând măsurătorile și achizițiile de date aferente, ar lua prea mult timp și ar fi (în majoritatea cazurilor) greoaie, vă propunem să realizați o simulare a experienței cu ajutorul calculatorului.

Programul ar putea fi structurat pe două părți:

- 1) măsurarea (simulată) a sarcinii electrice a unei picături de ulei;
- 2) reprezentarea grafică a rezultatelor simulării unor măsurători în funcție de mărimea sarcinii electrice a picăturilor de ulei.

Elevii vor analiza rezultatele diferitelor măsurători pentru variațiile date parametrilor.

Pentru o mai bună înțelegere, calculatorul va genera un set de date asupra cărora elevii se vor concentra, pentru a descoperi parametrii optimi pentru o bună prezentare grafică a acestora printr-o histogramă, observând distribuția regulată a zonelor care se formează.

Se va constata că valorile pentru viteză, respectiv pentru sarcina electrică, se grupează în jurul anumitor valori care sînt multiplii întregi ai celei mai mici valori, considerată sarcina electrică elementară (a electronului).

### Tema nr. 15:

Program pentru studiul mișcării unui corp aruncat în câmp gravitațional, în vid, cu viteza  $\vec{v}_0$ , ce formează un unghi  $\alpha$  cu orizontala.



Inițial, se prezintă pe ecran într-un mod sugestiv ecuațiile mișcării pentru cele două axe, stabilindu-se ecuația traiectoriei și parametrii mișcării (înălțimea maximă, timpul total al mișcării, distanța maximă pe orizontală).

Programul va permite ilustrarea graficului mișcării și componentelor vectorului viteză, respectiv verificarea dependenței parametrilor mișcării de unghiul  $\alpha$  și de valoarea  $\vec{v}_0$ , facilitând dezvoltarea unor considerații asupra conservării energiei mecanice.

#### *Tema nr. 16:*

Program pentru studiul compunerii oscilațiilor paralele (oscilații mecanice, curent alternativ, unde electromagnetice).

Se ilustrează problema solicitării simultane a unui punct al mediului, de către două forțe de tip elastic, care au aceeași frecvență. Se prezintă cazul compunerii oscilațiilor care au amplitudini egale, dar frecvențe diferite și generează fenomenul „bătăilor”. Elevii vor înțelege modul în care se mișcă un punct material solicitat simultan de două forțe oscilatorii de frecvențe diferite.

#### *Tema nr. 17:*

Program pentru studiul compunerii oscilațiilor perpendiculare, de aceeași frecvență, rezultând traiectorii de tip elipsă (cu cazurile particulare dreaptă și cerc), sau de frecvențe diferite, obținind „figurile Lissajous”. Se va studia dependența traiectoriei de defazaj.

### *Chimie*

#### *Tema nr. 18:*

Programe pentru consolidarea cunoștințelor despre legături chimice:

- legătura ionică, transferul de electroni, noțiunile de ioni pozitivi și negativi, exemplificarea pe ecran a modului de formare a legăturii la substanțele ionice;

- legătura covalentă, noțiunile de orbital atomic, orbital molecular de legătură, legăturile de tip s-s, p-p și s-p, molecule polare și nepolare, ilustrare pe ecran;

- hibridizarea și geometria moleculelor, noțiunea de orbital hibrid, hibridizarea  $sp^3$ ,  $sp^2$  și  $sp$ , vizualizare pe ecran;

- legătura coordinativă, atomi donori și acceptori, ioni poliatomici, combinații complexe, ilustrare pe ecran.

Se vor da exemplificări și se vor propune pentru rezolvare diverse exerciții de dezvoltare a capacității de a analiza, compara și interpreta observațiile.

Proprietățile moleculelor diatomice formate prin legături covalente pot fi ilustrate pentru modelele simple ale moleculei ion de hidrogen ( $H_2^+$ ) și ale moleculei de hidrogen. Cu ajutorul microcalculatorului se pot calcula și afișa grafic variațiile energiei cinetice și potențiale, respectiv variațiile densității sau distribuției electronice asociate formării legăturii covalente.

Se pot explica grafic creșterea energiei cinetice și descreșterea energiei potențiale la distanța de echilibru dintre nuclee și se poate ilustra contracția orbitalului atomic.



Se vor trasa curbele de densitate constantă (pentru  $H_2^+$ , se pot defini 4 curbe), ținând seama de principiul superpoziției, fără fenomenele de interferență și accelerare, iar apoi se vor trasa comparativ pe ecran curbele, luând în considerare și fenomenele menționate. Pentru molecula de hidrogen se va considera și fenomenul de corelație electronică.

Variația parametrilor de calcul și afișaj, oferă elevilor posibilitatea de a studia în profunzime procesele din chimia cuantică.

### 16.6.5. Teme pentru elaborarea unor programe aplicative

#### *Electronică-radiotehnică*

##### *Tema nr. 19:*

##### **Programe pentru calculul amplificatoarelor.**

Se vor considera diferite scheme cu tranzistoare, rezultând tipuri de probleme.

Pentru amplificatoarele cu tranzistoare se vor include, opțional, variante bazate pe modelul hibrid (cu parametrii  $h$  sau Giacoletto) al elementului activ pentru determinarea punctului static de funcționare a elementelor de circuit (divizorul de tensiune pentru polarizare, rezistența de sarcină, rezistența din emitor, condensatoarele de intrare, ieșire din emitor), calcularea amplificării de tensiune și de curent, a frecvențelor limită ale benzii de lucru. Se vor trasa opțional diagramele Bode și de fază.

Elevii vor învăța să lucreze cu scheme echivalente în c.c. și în c.a.

Se vor calcula câștigul în bucla deschisă a amplificatorului de bază, fără reacție, apoi factorul de transfer al rețelei de reacție, câștigul pe bucla de reacție, câștigul în bucla închisă și rezistențele de intrare și ieșire în prezența reacției negative.

De asemenea, de exemplu pentru un amplificator la care se dau amplificarea fără reacție, semnalul de intrare, puterea la ieșire și distorsiunea la nivelul armonicii a 2-a, considerând o reacție negativă de tip tensiune-serie (exprimată în dB), se pot determina semnalul de intrare necesar și valoarea distorsiunii armonice, exprimată în procente, astfel încât să se mențină puterea de ieșire la același nivel.

##### *Tema nr. 20:*

Program pentru învățarea și verificarea aptitudinilor pentru emisia și recepția de semnale în cod Morse.

Se vor verifica corectitudinea și viteza de lucru a elevului, care se poate antrena prin creșterea gradată a complexității textului și vitezei de emisie sau recepție (a se vedea programul „Morse” de pe casetă).

#### *Astronomie*

##### *Tema nr. 21:*

Program pentru calculul și afișarea efemeridei planetelor.

Efemerida unei planete este un tabel cu coordonatele sferice ale planetei (în sistemul de referință geocentric ecuatorial), date la intervale constante, în



cadrul unei perioade de timp anumit (o lună, un an, etc.).

Intervalul constant se numește pasul efemeridei. Determinarea acestor coordonate permite plasarea pozițiilor pe un atlas ceresc și căutarea planetei, cometei sau asteroidului în vederea observării sale.

Datele de intrare pentru program includ:

- constante universale;
- date relative la timp;
- elementele orbitale ale planetei;
- elementele orbitale ale Pământului.

Un subprogram va citi și transforma datele de intrare (grade, minute și secunde în radiani și data calendaristică în ziua iuliană modificată), iar un alt subprogram va afișa datele.

Programul principal va efectua:

- calculul anomaliei mijlocii  $M = n(t - t_0)$  pentru planeta respectivă și pentru Pământ;
- rezolvarea ecuației lui Kepler pentru planetă și Pământ, determinând anomalile excentrice ale acestora;
- calculul coordonatelor heliocentrice orbitale ale planetei și ale Pământului;
- rotirea sistemului de axe (de unghiuri  $-\omega$ ,  $-i$  și  $-\Omega$ ), pentru planetă și pentru Pământ, obținând coordonatele heliocentrice ecliptice;
- translația sistemului de referință, aducând originea sistemului în centrul Pământului, obținând coordonatele geocentrice ecliptice ale planetei;
- trecerea la sistemul de referință geocentric ecuatorial, printr-o rotație de unghi  $-\epsilon$  în jurul axei  $Ox$ ;
- trecerea de la coordonatele carteziene geocentrice ecuatoriale la coordonatele sferice din același sistem, obținând ascensia dreaptă și declinația;
- transformarea declinației din radiani în grade, minute și secunde, iar ascensia în ore, minute și secunde de timp;
- transformarea timpului, în data calendaristică și ora, pentru afișare;
- afișarea (tipărirea) momentului, ascensiei drepte și declinației.

Prin această temă, sugerăm realizarea programelor în BASIC HC-85.

Ca o ilustrare concretă a ceea ce se poate realiza în cadrul cluburilor de informatică în capitolul 19 se face și prezentarea generală a celor 15 micro-pachete de programe care sînt incluse pe casetele ce însoțesc această lucrare.

---

a. Unele obiective ale temelor propuse și discutate mai sus (de pildă ARP, GRAF2, TREP, TRIGRAF), sînt similare celor rezolvate de Mihai Andrei Mărșanu în 1985, ca elev în clasa a XI-a la Liceul Mihai Viteazul din București, în cadrul sistemului CAN de 12 programe pentru studierea matematicii. Sistemul CAN a fost premiat la primul Concurs național de programare organizat de revista Știință și Tehnică și intitulat „Aplicațiile calculatorului în societate”.



## Partea a VIII-a

# PROGRAME EDUCAȚIONALE ÎN BASIC PE CALCULATORUL HC-85

## Capitolul 17. | Programe pentru clasele I-VIII \*)

Conținutul acestui capitol integrează în preocupările învățătorilor și ale profesorilor o serie de teme și titluri de programe, sau secvențe ale acestora, necesare procesului de predare-învățare, îndeosebi activităților în cluburi. O parte din programe sînt precedate de schemele logice corespunzătoare.

Tematica de față se adresează mai multor discipline școlare ale claselor I—VIII. O bună parte din teme solicită o îndrumare în pași, necesară alcătuirii schemei logice — aceasta fiind o călăuză teoretică și practică sigură a pregătirii pentru scrierea de programe în BASIC.

Pentru reușita acestei acțiuni se pot folosi indicațiile și exemplele tratate în capitolul 15. Este bine ca în timpul lucrului să se accentueze ideea că schema logică pentru rezolvarea unei probleme nu este unică, încurajîndu-se astfel realizarea individuală a acestor scheme, din care să se selecteze cele mai bune, explicîndu-se neajunsurile celorlalte. Este, de asemenea, necesar să se lămurească prin exemple și ideea că nici programul de rezolvare a unei probleme nu este unic, folosindu-se acest fapt pentru stimularea elevilor în abordarea originală a problemelor tratate. Vezi și lucrul în pseudocod (ANEXE).

În lucrările pentru calculator prezentate în acest capitol există o serie de programe cu caracter mai general, care se folosesc la rezolvarea unor anumite clase de probleme. Există însă și programe sau secvențe de program, necesare procesului de predare, specifice unui singur tip de problemă, ca de exemplu: trasările de grafice ale unor tipuri de funcții elementare sau locuri geometrice, prezentarea de imagini specifice unor anumite lecții din programa școlară, exerciții, îndeosebi teme pentru cercurile de elevi. Unele secvențe de programe nu sînt încheiate, ele putînd fi dezvoltate de elevi. (v. și § 16.6).

În general, scopul prezentării acestor secvențe, sau a programelor scurte, este acela de a oferi un material cu care se poate interveni fie în procesul de



predare, fie în procesul de consolidare sau de control al însușirii unor noțiuni — pentru care solicitarea calculatorului devine eficientă. În plus, aceste programe scurte pot deveni titluri de lecție practică de informatică și pentru elevii claselor IX—X. Ele constituie însă teme pentru cluburile școlare ale claselor I—VIII — în munca lor cu calculatorul. Aceste teme sînt, în general, axate pe durata unei ore de muncă în laborator sau în clasă, sub îndrumarea profesorului, sau în activitatea independentă a elevilor. Le vom numi: „FIȘA-PROGRAM”. Desigur, ele sînt facultative.

Atît profesorii sau îndrumătorii de cercuri, cît și elevii acestora, pot folosi aceste programe ca ghid în experiența muncii lor de început. În etapele caracterizate prin grafism mai complicat, sprijinul profesorului îndrumător este absolut necesar chiar dacă elevul are pe masă fișa-program. După cîștigarea unei mai bune experiențe în munca de programare și de folosire a calculatorului, se poate păși în stadiul activității creatoare de alcătuire dirijată sau independentă a unor scheme și programe în BASIC, din diversele discipline școlare și din producție. În cadrul activității de club ca — spre exemplu — în cluburile la care se folosesc locuri geometrice, sau proprietățile funcțiilor și ale relațiilor cu *modul*, elevii pot fi solicitați să se întrecă în a intui și oferi soluții grafice înainte de afișarea lor pe ecran.

Există însă și unele programe, ca — spre exemplu — cele de vizualizare a unor locuri geometrice, al căror suport teoretic solicită cunoașterea unor noțiuni de algebră și geometrie analitică. Ele pot fi folosite în ceea ce privește partea analitică, numai la cluburile pentru avansați. Dar, demonstrația sintetică și îndeosebi vizualizarea grafică, în mișcare, a modului în care este generat locul geometric, constituie un element necesar și compatibil vîrstei școlarilor din clasele VI—VIII.

De asemenea, în scopul dezvoltării la elevi a unor calități care permit rapiditate în luarea unor decizii, precum și pregătirea în perspectivă a spiritului de programare și de investigație creatoare al elevilor, pot fi folosite o serie de jocuri compatibile vîrstei școlarilor.

Utilizînd jocuri educative cunoscute, elevii pot ajunge la crearea de noi situații de joc, sau chiar la crearea unor jocuri noi. Se realizează astfel climatul învățării prin acțiune — proces care conduce la stimularea creativității elevilor, a capacității lor de gîndire, decizie și acțiune, a interesului și pasiunii pentru știința informaticii și aplicațiile acesteia. Problemele pot crea o serie de situații de joc matematic cu conținut geometric și de grafică din această lucrare.

---

\* Programele din Capitolul 17 și primele 40 de programe din Capitolul 18 sînt lucrate cu profesorii și elevii cercului de informatică de la Liceul „Dimitrie Cantemir” București, în perioada 1984 — 1986. Menționăm, îndeosebi activitatea elevilor: Petrescu Iacob, Bălan Radu și Tudor Sorin.



## 17.1. Program „ADUNAREA”

1. Tema: Adunarea a două numere naturale date A, B.
2. Schema logică
3. Program BASIC (HC-85) — ADUNAREA
- 4) OBSERVAȚII: pentru a salva programul pe casetă se procedează astfel:
  - a) Fără a pune număr de linie tastezi S
  - b) Tastezi „ADUNAREA”
  - c) Tastezi ENTER și apare mesajul: START TAPE THEN PRESS ANYKEY
  - d) Porniți casetofonul pe înregistrare.
  - e) Tastezi orice și programul va fi salvat pe casetă.

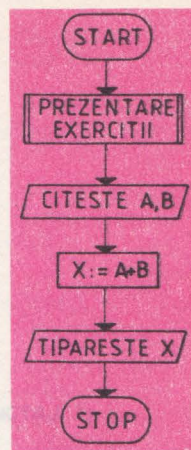


Fig. 17.1. Schema logică „Adunarea”

```

1 REM 1.Program "ADUNAREA"
2 REM 1'. Tema: Adunarea a doua
3 REM numere date A si B
4 REM 2'. Program BASIC (HC-85)
10 CLS
20 PRINT "Program de aflare a sumei ";
30 PRINT "a doua numere A si B"
40 GO SUB 140
50 CLS
60 PRINT "Introduceti numerele A, B:"
70 INPUT "A=";A;" B=";B
80 PRINT "A=";A;" B=";B
90 LET X=A+B
100 PRINT "A=";A;" B=";B;" X=";X
110 PRINT "REZULTATUL ESTE X=";X
120 GO SUB 220
  
```

```

130 GO TO 50
140 PRINT "DAM CITEVA EXEMPLE REZOL";
150 PRINT "VATE DE CAL";
160 PRINT "CALCULATOR:"
170 PRINT AT 10,10;" 1) 2+ 4= 6"
180 PRINT AT 11,10;" 2) 34+21=55"
190 PRINT AT 12,10;" 3) 45+12=57"
200 PRINT AT 13,10;" 4) 2+ 1= 3"
210 PRINT AT 14,10;" 5) 22+56=78"
220 PRINT "Daca doriti sa rezolvati";
230 PRINT "exer- citii cu calculato";
240 PRINT "rul apasati tasta A"
250 PAUSE 0
260 IF INKEY$="a" THEN RETURN
270 IF INKEY$="A" THEN RETURN
280 STOP
  
```

## 17.2. Program test — Adunarea

Tema: Test „Adunare”.

```

1 REM 1.Program "Test Adunare"
2 REM 1'. Tema: Test adunare
3 REM 2'. Program BASIC (HC-85)
10 CLS
20 PRINT "ADUNAREA CU TRECERE PESTE";
30 PRINT "ORDIN A DOUA NUMERE";
40 PRINT "NATURALE MAI MICI CA";
50 PRINT "100, AVIND SUMA ";
60 PRINT "CEL PUTIN 100"
70 PRINT "PROGRAMUL CONTINE TREI ";
80 PRINT "PARTI: -1. NUMERE FOR";
90 PRINT "MATE NUMAI DIN ZECI, ";
100 PRINT "AVIND SUMA 100 -2. ";
110 PRINT "NUMERE FORMATE NUMAI ";
120 PRINT "DIN ZECI, AVIND SUMA MAI";
130 PRINT "MARE DECIT 100 ";
140 PRINT "-3. NUMERE ";
150 PRINT "MAI MICI DECIT 100, ";
160 PRINT "AVIND SUMA MAI MARE ";
170 PRINT "DECIT 100"
180 INPUT "ALEGETI UNA DINTRE PARTI: ";N
190 IF N=1 THEN GO TO 230
200 IF N=2 THEN GO TO 650
210 IF N=3 THEN GO TO 1130
  
```

```

220 GO TO 160
230 CLS
240 PRINT "NUMERE FORMATE NUMAI DIN ";
250 PRINT "ZECI, AVIND SUMA 100";
260 PRINT "INDICATI NUMARUL DE EXER";
270 PRINT "CITII CE VOR FI TES";
280 PRINT "TATE": INPUT X
290 IF X=0 OR X=1 THEN GO TO 260
300 LET K=1: LET P=0
310 CLS: IF K=X+1 THEN GO TO 1630
320 PRINT AT 5,2;"EXERCITIUL ";K
330 PRINT AT 3,15;"PUNCTAJ ";P;" PUNCTE"
340 LET S=INT (RND*10)*10
350 PRINT AT 7,5;100-S;"+";S;"=";
360 INPUT "RASPUIS ";R: PRINT R
370 IF R=100 THEN GO TO 1690
380 PRINT AT 9,10;"RASPUIS GRESIT!"
390 PRINT AT 11,5;"PERSEVEREAZA SI VEI";
400 PRINT "REUSII"
410 PRINT AT 5,15;"RASPUISUL CORECT:"
420 PRINT AT 7,19;100-S;"+";S;"=100"
430 PRINT FLASH 1;AT 18,0;"TASTEAZA ";
440 PRINT FLASH 1;"ORICE PENTRU A ";
450 PRINT FLASH 1;"CONTINUA"
  
```



```

460 LET K=K+1
470 PAUSE 6000: GO TO 310
480 CLS : LET T=(P/(K-1))*100
490 PRINT AT 3,5;"REZULTATUL TESTULUI:"
500 PRINT AT 8,5;"*-DIN ";K-1;" EXER";
510 PRINT "CITII PROPUSE PENTRU RE";
520 PRINT "ZOLVARE LA ";P;" S-A ";
530 PRINT " RASPUNS CORECT"
540 PRINT AT 11,5;"*-PROCENTAJUL RAS";
550 PRINT "PUNSURILOR CORECTE";
560 PRINT "ESTE DE ";T;" LA SUTA"
570 PRINT "PENTRU A REVENI LA CUPRINS";
580 PRINT " TASTATI C. DACA DO";
590 PRINT "RITI CA PROGRAMUL SA SE";
600 PRINT " RULEZE IN CONTI";
610 PRINT "NUARE TASTATI Z"
620 INPUT INKEY$
630 IF INKEY$="C" THEN GO TO 10
640 IF INKEY$="c" THEN GO TO 10
650 IF INKEY$="" THEN GO TO 630
660 CLS
670 PRINT " NUMERE FORMATE NUMAI DIN ";
680 PRINT "ZECI, AVIND SUMA MAI MA";
690 PRINT "RE DECIT 100 "
700 PRINT " INDICATI NUMARUL DE EXER";
710 PRINT "CITII CE VOR FI TES";
720 PRINT "TATE": INPUT X
730 IF X=0 OR X=1 THEN GO TO 260
740 LET K=0: LET P=0
750 CLS : LET K=K+1
760 IF K=X+1 THEN GO TO 1750
770 PRINT AT 5,2;"EXERCITIUL ";K
780 PRINT AT 3,15;"PUNCTAJ ";P;" PUNCTE"
790 LET S=INT (RND*10)*10
800 LET W=INT (RND*10)*10
810 PRINT AT 7,5;S;"+";W;"=";
820 INPUT "RASPUNS ";R: PRINT R
830 IF R=S+W THEN GO TO 1810
840 PRINT AT 9,10;"RASPUNS GRESIT!?"
850 PRINT AT 11,5;"PERSEVEREAZA SI VEI ";
860 PRINT " REUSII"
870 PRINT AT 5,15;"RASPUNSUL CORECT:"
880 PRINT AT 7,19;S;"+";W;"=";
890 PRINT S+W: PRINT AT 18,0;"TASTEAZA";
900 PRINT " ORICE PENTRU A CONTINUA"
910 PAUSE 6000: GO TO 750
920 CLS : LET T=(P/(K-1))*100
930 PRINT AT 3,5;"REZULTATUL TESTULUI:"
940 PRINT AT 8,5;"*-DIN ";K-1;
950 PRINT " EXERCITII PROPUSE";
960 PRINT " PENTRU REZOLVARE,LA ";
970 PRINT P;" S-A RASPUNS ";
980 PRINT "CORECT"
990 PRINT AT 11,5;
1000 PRINT AT 11,5;"*-PROCENTAJUL";
1010 PRINT " RASPUNSURILOR ";
1020 PRINT "CORECTE ESTE DE ";T;
1030 PRINT " LA SUTA"

```



```

1040 PRINT "PENTRU A REVENI LA CUPRINS";
1050 PRINT "TASTATI C. DACA ";
1060 PRINT "DORITI CA PROGRAMUL SA ";
1070 PRINT "SE RULEZE IN CONTINUARE ";
1080 PRINT "TASTATI Z"
1090 INPUT INKEY$
1100 IF INKEY$="C" THEN GO TO 10
1110 IF INKEY$="c" THEN GO TO 10
1120 IF INKEY$="" THEN GO TO 1100
1130 CLS
1140 PRINT "NUMERE MAI MICI DECIT ";
1150 PRINT "100,AVIND SUMA MAI MARE DE 100"
1160 PRINT "INDICATI NUMARUL DE ";
1170 PRINT "EXERCITII CE VOR ";
1180 PRINT "FI TESTATE": INPUT X
1190 IF X=0 OR X=1 THEN GO TO 1160
1200 LET K=0: LET P=0
1210 CLS : LET K=K+1:
1220 IF K=X+1 THEN GO TO 1880: PRINT AT 3,15;"PUNCTAJ ";
1230 PRINT AT 5,2;"EXERCITIUL ";K:
1240 PRINT AT 3,15;"PUNCTAJ ";P;" PUNCTE"
1250 LET S=INT (10*RND*10)
1260 LET W=INT (10*RND*10)
1270 PRINT AT 7,5;S;"+";W;"=";:
1280 INPUT "RASPUNS ";R: PRINT R
1290 IF R=S+W THEN GO TO 1310
1300 GO TO 1370
1310 PRINT AT 9,10;"RASPUNS CORECT!"
1320 LET P=P+1
1330 PRINT FLASH 1;AT 18,0;"TASTEAZA";
1340 PRINT FLASH 1;" ORICE PENTRU A ";
1350 PRINT FLASH 1;"CONTINUA"
1360 PAUSE 5000: GO TO 1210
1370 PRINT AT 9,10;"RASPUNS GRESIT!?"
1380 PRINT AT 11,5;"PERSEVEREAZA SI VEI";
1390 PRINT " REUSI!";
1400 PRINT AT 5,15;"RASPUNSUL CORECT:"
1410 PRINT AT 7,19;S;"+";W;"=";S+W
1420 PRINT FLASH 1;AT 18,0;"TASTEAZA";
1430 PRINT FLASH 1;" ORICE PENTRU A ";
1440 PRINT FLASH 1;"CONTINUA"
1450 PAUSE 6000: GO TO 1210
1460 CLS : LET T=(P/(K-1))*100
1470 PRINT AT 3,5;"REZULTATUL TESTULUI:"
1480 PRINT AT 8,5;"*-DIN ";K-1;
1490 PRINT " EXERCITII PROPUSE ";
1500 PRINT "PENTRU REZOLVARE,LA ";
1510 PRINT P;" S-A RASPUNS CORECT"
1520 PRINT AT 11,5;"*-PROCENTAJUL ";
1530 PRINT "RASPUNSURILOR CORECTE ";
1540 PRINT "ESTE DE ";T;" LA SUTA"
1550 PRINT "PENTRU A REVENI LA CUPRINS";
1560 PRINT "TASTATI C."
1570 INPUT INKEY$
1580 IF INKEY$="C" THEN GO TO 10
1590 IF INKEY$="c" THEN GO TO 10
1600 IF INKEY$="" THEN GO TO 1580
1610 CLS
1620 STOP

```



```

1630 PRINT AT 3,15;"PUNCTAJ ";P;
1640 PRINT " PUNCTE"
1650 PRINT FLASH 1;AT 18,0;"TASTEAZA ";
1660 PRINT FLASH 1;"ORICE PENTRU A ";
1670 PRINT FLASH 1;"CONTINUA"
1680 PAUSE 6000: GO TO 480
1690 PRINT AT 9,10;"RASPUNS CORECT!"
1700 LET P=P+1: LET K=K+1
1710 PRINT FLASH 1;AT 18,0;"TASTEAZA ";
1720 PRINT FLASH 1;"ORICE PENTRU A ";
1730 PRINT FLASH 1;"CONTINUA"
1740 PAUSE 6000: GO TO 310
1750 PRINT AT 3,15;"PUNCTAJ ";P;
1760 PRINT " PUNCTE"
1770 PRINT FLASH 1;AT 18,0;"TASTEAZA ";
1780 PRINT FLASH 1;"ORICE PENTRU A ";
1790 PRINT FLASH 1;"CONTINUA"
1800 PAUSE 6000: GO TO 920
1810 PRINT AT 9,10;"RASPUNS CORECT!"
1820 LET P=P+1
1830 PRINT FLASH 1;AT 18,0;"TASTEAZA ";
1840 PRINT FLASH 1;"ORICE PENTRU A CON";
1850 PRINT FLASH 1;"TINUA"
1860 PAUSE 6000: GO TO 750
1870 STOP
1880 PRINT AT 3,15;"PUNCTAJ ";
1890 PRINT P;" PUNCTE"
1900 PRINT FLASH 1;AT 18,0;"TASTEAZA ";
1910 PRINT FLASH 1;"ORICE PENTRU A CON";
1920 PRINT FLASH 1;"TINUA"
1930 PAUSE 6000: GO TO 1460

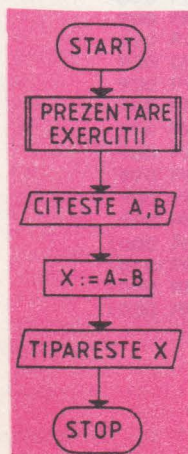
```

### 17.3. Program „SCĂDEREA”

1. Tema: Program de scădere a două numere naturale date A, B, unde  $A > B$ .

2. Schema logică

3. Program BASIC (HC-85)



```

1 REM .Program 'SCADEREA'
2 REM 1'. Tema: Scaderea a doua
3 REM numere date A si B
4 REM 2'. Program BASIC (HC-85)
10 CLS
20 PRINT "Program de aflare a dife";
30 PRINT "rentei a doua numere A";
40 PRINT " si B": GO SUB 140
50 CLS
60 PRINT ""Introduceti numerele A, B:"
70 INPUT "A=";A;" B=";B
80 PRINT "A=";A;" B=";B
90 LET X=A-B
100 PRINT "A;"-";B;"=";X
110 PRINT ""REZULTATUL ESTE X=";X
120 GO SUB 220
130 GO TO 50
140 PRINT ""DAM CITEVA EXEMPLE RE";
150 PRINT "ZOLVATE DE CAL";
160 PRINT "CULATOR:"
170 PRINT AT 10,10;"1) 15- 7= 6"
180 PRINT AT 11,10;"2) 8- 1= 7"
190 PRINT AT 12,10;"3) 13- 0=13"
200 PRINT AT 13,10;"4) 34-21=13"
210 PRINT AT 14,10;"5) 56-22=34"
220 PRINT ""Daca doriti sa rezolvati";
230 PRINT " exer- citii cu calculato";
240 PRINT "rul apasati tasta S"
250 PAUSE 0
260 IF INKEY$="s" THEN RETURN
270 IF INKEY$="S" THEN RETURN
280 STOP

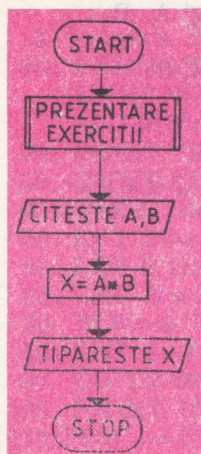
```

Fig. 17.2. Schema logică „Scăderea”



## 17.4. Program „ÎNMULTIREA”

1. Tema: Înmulțirea a două numere naturale A, B.
2. Schema logică
3. Program BASIC (HC-85).



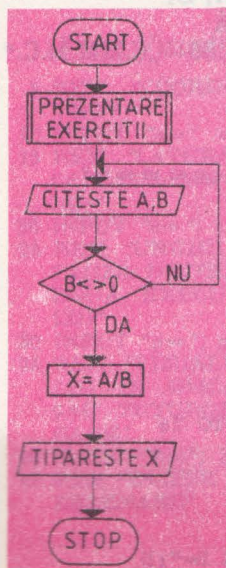
```

1 REM 3. Program 'INMULTIREA'
2 REM 1'. Tema: Inmultirea a doua numere date A si B
3 REM numere date A si B
4 REM 2'. Program BASIC (HC-85)
10 CLS
20 PRINT "Program de aflare a produ";
30 PRINT "sului      a doua numere A,B"
40 GO SUB 140
50 CLS
60 PRINT "'Introduceti numerele A si B:"
70 INPUT "A=";A;" B=";B
80 PRINT "'A=";A;" B=";B
90 LET X=A*B
100 PRINT "'A;"*";B;"=";X
110 PRINT "'REZULTATUL ESTE X=";X
120 GO SUB 220
130 GO TO 50
140 PRINT "'DAM CITEVA EXEMPLE REZOL";
150 PRINT "VATE DE          CALCU";
160 PRINT "LATOR:"
170 PRINT AT 10,10;"1) 12* 2= 24"
180 PRINT AT 11,10;"2) 23* 4= 92"
190 PRINT AT 12,10;"3) 3* 123= 369"
200 PRINT AT 13,10;"4) 1*1234=1234"
210 PRINT AT 14,10;"5) 13* 0= 0"
220 PRINT "'Daca doriti sa rezolvati";
230 PRINT " exer- citii cu calcula";
240 PRINT "torul apasati tasta M"
250 PAUSE 0
260 IF INKEY$="m" THEN RETURN
270 IF INKEY$="M" THEN RETURN
280 STOP
  
```

Fig. 17.3. Schema logică „Înmulțirea”

## 17.5. Program „IMPĂRȚIREA”

1. Tema: Împărțirea a două numere naturale A, B.
2. Schema logică
3. Program BASIC (HC-85)



```

1 REM 4. Program 'IMPARTIREA'
2 REM 1'. Temă: Impartirea a doua
3 REM numere date A si B
4 REM 2'. Program BASIC (HC-85)
10 CLS
20 PRINT "Program de aflare a citului";
30 PRINT "      a doua numere A si B"
40 GO SUB 160
50 CLS
60 PRINT "'Introduceti numerele A, B:"
70 INPUT "A=";A;
80 INPUT " B=";B
90 IF B=0 THEN GO SUB 340
100 PRINT "'A=";A;" B=";B
110 LET X=A/B
120 PRINT "'A;"*";B;"=";X
130 PRINT "'REZULTATUL ESTE X=";X
140 GO SUB 270
150 GO TO 50
160 PRINT "'DAM CITEVA EXEMPLE RE";
170 PRINT "ZOLVATE DE          CAL";
180 PRINT "CULATOR:"
190 PRINT AT 10,10;"1) 12: 4= 3"
200 PRINT AT 11,10;"2) 23: 0=IMPO";
210 PRINT "SIBIL"
220 PRINT AT 12,10;"3) 23: F=";
230 PRINT "Variable "
240 PRINT AT 13,22;"not found"
250 PRINT AT 14,10;"4) 34:17= 2"
260 PRINT AT 15,10;"5) 1234:21= 58.7619"
270 PRINT "'Daca doriti sa rezolvati";
280 PRINT " exer- citii cu calculato";
290 PRINT "rul apasati tasta P"
300 PAUSE 0
310 IF INKEY$="p" THEN RETURN
320 IF INKEY$="P" THEN RETURN
330 STOP
340 PRINT "' FLASH 1;" Aveti grija!";
350 PRINT " FLASH 1;" B<>0!! "
360 GO TO 80
  
```

Fig. 17.4. Schema logică „Împărțirea”



## 17.6. Program „C.M.M.D.C.”

1. Tema: Calcularea c.m.m.d.c. a două numere întregi A, B.
2. SCHEMA LOGICĂ
3. Program BASIC HC-85

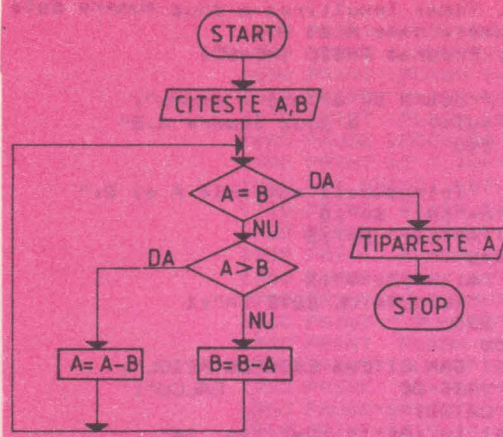


Fig. 17.5. Schema logică „c.m.m.d.c.”

```

5 PRINT "PROGRAM BASIC HC-85 CMMDC "
1. PAUSE 200:CLS
2. PRINT "CEL MAI MARE DIVIZOR COMUN"
3. INPUT "DATI CELE DOUA NUMERE A=" ; A
4. INPUT "B=";B
5. PRINT "A=";A; PRINT "B=";B
55 IF INT A<> A OR INT B<> B THEN GO TO 1.
6. IF A=B THEN GO TO 1.
7. IF A>B THEN GO TO 9.
8. LET B=B-A;GO TO 6.
9. LET A=A-B; GO TO 6.
10. PRINT "###"; PRINT "C.M.M.D.C.=";A;" ###"
11. STOP
  
```

## 17.7. Program „C.M.M.M.C.”

1. Tema: Calcularea c.m.m.m.c. a două numere întregi A, B.
2. SCHEMA LOGICĂ
3. Program BASIC HC85—„C.MM.M.C.”

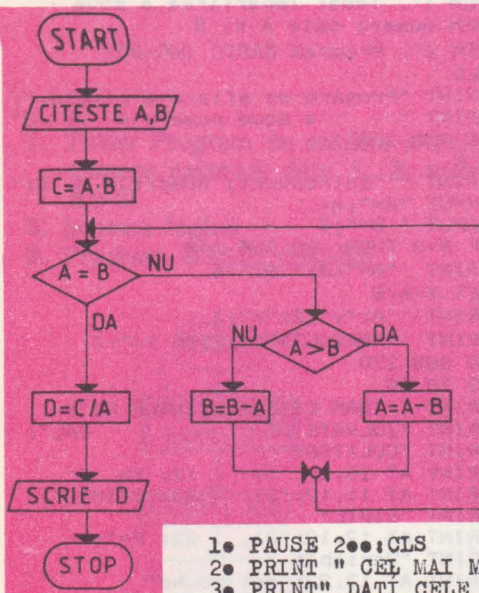


Fig. 17.6. Schema logică „c.m.m.m.c.”

```

1. PAUSE 200:CLS
2. PRINT "CEL MAI MIC MULTIPLU COMUN"
3. PRINT" DATI CELE DOUA NUMERE"
4. INPUT A,B;LET C=A*B
45 IF INT A<> A OR INT B<> B THEN GO TO 1.
6. IF A=B THEN GO TO 9.
65 IF A>B TEHN GO TO 8.
7. LET B=B-A; GO TO 6.
8. LET A=A-B; GO TO 6.
9. LET D=C/A
100 PRINT "CEL MAI MIC MULTIPLU COMUN ESTE D=";D
11. STOP
  
```



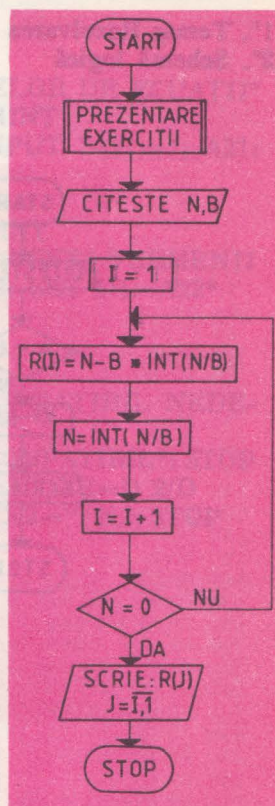
## 17.8. Program „CONVERSIE”

1. Tema: Conversia numerelor naturale din baza 10 într-o bază mai mică
2. SCHEMA LOGICĂ

```

3. Program BASIC HC-85 — "CONVERSIE"
10. PRINT "PROGRAM DE TRANSFORMARE A
    UNUI NUMĂR DIN BAZA 10 ÎNTR-O BAZĂ
    MAI MICĂ"
20. INPUT "INTRODUCETI NUMĂRUL N"; N
30. INPUT "INTRODUCETI BAZA B"; B
40. DIM R (100)
50. LET I=1
60. LET R (I) =N-B * INT (N/B)
65. LET N=INT (N/B)
70. IF N=0 THEN GO TO 100
80. LET I=I+1
90. GO TO 60
100. PRINT "N="
110 FOR J=I TO 1 STEP -1
120 PRINT R (J)
130. NEXT J
140. PRINT "DACĂ DORIȚI SĂ CONTINUAȚI,
    TASTAȚI P; DACĂ NU, TASTAȚI ORICE"
150. PAUSE 0
160. IF INKEY $="P" THEN GO TO 20
170. STOP
    
```

Fig. 17.7. Schema logică „Conversia”



## 17.30 PROGRAM „ROTIREA UNUI PĂTRAT”

Tema: Vizualizarea rotirii unui pătrat.

```

5 REM "COORDONATELE UNUI
  VÂRF"
10 INPUT "a=" ;a "b="; b
15 REM "LATURA PĂTRATULUI ȘI
  CAPĂȚUL SUPERIOR AL CICLU-
  LUI"
20 INPUT "c="; c, "u="; u
25 REM "UNGHIUL DE ROTAȚIE ȘI
  PAUZA DINTRE
  DOUĂ EXECUȚII ALE DESENU-
  LUI"
30 INPUT "p="; p, "x="; x
35 REM "VIZUALIZAREA PĂTRATU-
  LUI ÎN MIȘCARE
  SAU TUTUROR CELOR ANTERI-
  OARE"
40 INPUT "Z="; Z
45 REM "ÎNCEPE CICLUL DE
  DESENARE"
50 FOR i=0 TO u STEP p
55 LET ui=i+PI/2
60 PLOT a, b
70 DRAW c * SIN i, c * COS i
80 DRAW c * SIN ui, c * COS ui
90 DRAW -c * SIN i, -c * COS i
100 DRAW -c * SIN ui, -c * COS ui
105 PAUSE X
110 DRAW OVER Z; c * SIN i, c * COS i
120 DRAW OVER Z; c * SIN ui, c * COS ui
130 DRAW OVER Z; -c * SIN i, -c * COS i
140 DRAW OVER Z; -c * SIN ui, -c * COS ui
155 REM "SFÎRSITUL UNUI CICLU"
160 NEXT i
    
```

## 17. PROGRAME CLASELE I-VIII



## 17.9. Program „EC.GR.I”

- 1°. Tema: Rezolvarea ecuației de gradul I cu o necunoscută,  $AX+B=0$ .
- 2°. Schemă logică

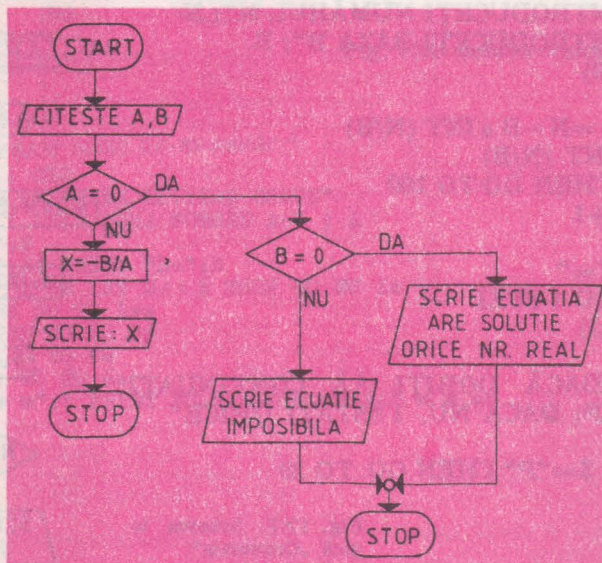


Fig. 17.8. Schema logică „Ec. gr. I”

- 3°. Program BASIC (HC-85) – ”ECUAȚIA GRAD I”
10. PRINT ”PROGRAM DE REZOLVAPE A ECUAȚIEI DE GRADUL I”
20. PRINT ”ECUAȚIA ESTE DE FORMA A \* X+B=0”
30. INPUT ”INTRODUCETI COEFICIENTII A, B; ” ; A, B
40. IF A=0 THEN GO TO 80
50. LET X = - B/A
60. PRINT ”SOLUȚIA ECUAȚIEI ESTE: X= ” ; X
70. STOP
80. IF B=0 THEN PRINT ”ECUAȚIA ADMITE SOLUȚIE ORICE NUMĂR REAL”: STOP
90. IF B <> 0 THEN PRINT ”ECUAȚIE IMPOSIBILĂ”: STOP



## 17.10. Program „SISTEM 2\*2”

- 1°. Tema: Rezolvarea unui sistem de două ecuații cu două necunoscute: (în cazul  $A_{12} \neq 0$  sau  $A_{22} \neq 0$ )
- 2°. Schema logică
- 3°. Program BASIC (HC-85) — „SISTEM 2\*2”
10. PRINT "PROGRAM DE REZOLVARE A SISTEMELOR DE ECUAȚII"
20. PRINT "' ' ' "INTRODUCETI COEFICIENTII": PRINT
30. INPUT "A11="; A11: INPUT "A12="; A12: INPUT "A21="; A21: INPUT "A22="; A22.
40. PRINT:PRINT "A11="; A11; "A12="; A12
50. PRINT:PRINT "A21="; A21; "A22="; A22: PRINT
60. LET D=A11 \* A22 - A21 \* A12: PRINT "INTRODUCETI TERMENII LIBERI" :INPUT "B1="; B1:INPUT "B2="; B2: LET D1=A22\*B1 - A12 \* B2
70. PRINT:PRINT "B1="; B1
80. PRINT:PRINT "B2="; B2 : PRINT
90. IF D=0 AND D1=0 THEN PRINT "SISTEM COMPATIBIL NEDETERMINAT": STOP
100. IF D=0 AND D1<>0 THEN PRINT "SISTEM INCOMPATIBIL": STOP
110. LET D2=A11 \* B2 - A21 \* B1:LET X1=D1/D :LET X2=D2/D
120. PRINT "SOLUȚIILE SÎNT X1="; X1;"; X2="; X2: STOP

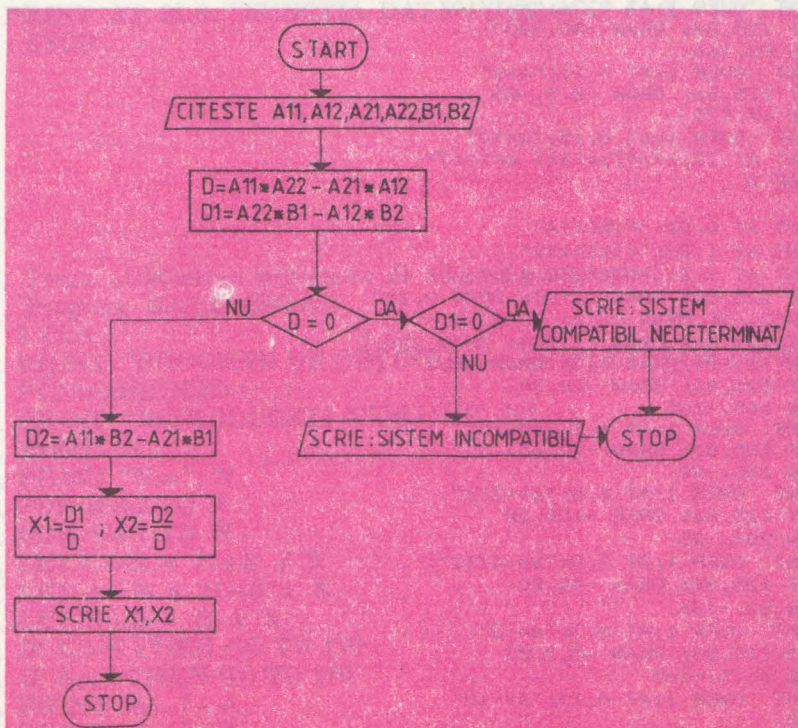


Fig. 17.9. Schema logică „Sistem 2\*2”



## 17.11. Program „FUNCȚIA”

1°. Tema: Trasarea graficului funcției  $X \rightarrow a * X$ ,  $X \in R$ ,  $a \in R$

2°. Program BASIC (HC-85).

```
1 REM 11) Program "FUNCȚIA"
2 REM 1°. Tema Subprogram de
3 REM trasare a graficului funcției
4 REM  $x \rightarrow a * x$ ,  $x \in R$ , pentru diferiți
5 REM coeficienți reali "a"
6 REM 2°. Program BASIC (HC-85)
10 PRINT AT 0,2;"FUNCȚIA";AT 2,3;"f(x)=a*x";AT 3,1;"GRAFICE a>0"
20 PRINT AT 0,2;"FUNCȚIA"
30 PRINT AT 2,3;"f(x)=a*x"
40 PRINT AT 3,1;"GRAFICE a>0"
50 PLOT 125,86: DRAW 0,-80
60 DRAW 0,160: DRAW 3,-3: DRAW -3,3:
70 DRAW -3,-3: PLOT 125,86: DRAW -110,0
80 DRAW 220,0: DRAW -3,-3: DRAW 3,3
90 DRAW -3,3
100 PLOT 125,86: DRAW -80,-80: DRAW 165,165
110 PRINT OVER 1;AT 1,26;"a=1"
120 PLOT 125,86: DRAW -90,-45: DRAW 180,90
130 PRINT OVER 1;AT 6,26;"a=1/2"
140 PLOT 125,86: DRAW -100,-10
150 DRAW 200,20
160 PRINT OVER 1;AT 9,26;"a=1/10"
170 PLOT 125,86: DRAW -40,-80
180 DRAW 83,166
190 PRINT OVER 1;AT 0,21;"a=2"
200 PLOT 125,86: DRAW -8.3,-83
210 DRAW 17,170
220 PRINT OVER 1;AT 0,15;"a=10"
230 PRINT AT 15,17;"Tastati orice!"
240 PAUSE 0
250 CLS
260 PRINT AT 0,22;"FUNCȚIA"
270 PRINT AT 1,23;"f(x)=a*x"
280 PRINT AT 3,21;"GRAFICE a<0"
290 PLOT 125,86: DRAW 0,-80: DRAW 0,160
300 DRAW 3,-3: DRAW -3,3: DRAW -3,-3
310 PLOT 125,86: DRAW -110,0: DRAW 220,0
320 DRAW -3,-3: DRAW 3,3: DRAW -3,3
330 PLOT 125,86: DRAW -80,80
340 DRAW 165,-165
350 PRINT OVER 1;AT 1,2;"a=-1"
360 PLOT 125,86: DRAW -90,45
370 DRAW 180,-90
380 PRINT OVER 1;AT 6,0;"a=-1/2"
390 PLOT 125,86: DRAW -100,10
400 DRAW 200,-20
410 PRINT OVER 1;AT 9,0;"a=-1/10"
420 PLOT 125,86: DRAW -40,80
430 DRAW 83,-166
440 PRINT OVER 1;AT 0,7;"a=-2"
450 PLOT 125,86: DRAW -8.3,83
460 DRAW 16.5,-165
470 PRINT OVER 1;AT 0,13;"a=-10"
480 STOP
```



## 17.12. Program „FUNCȚIA”

- 1°. Tema: Subprogram pentru trasarea graficului funcției  $f$ ,  
 $f(X) = X^2 - |X|$ ,  $X \in \mathbb{R}$ .
- 2°. Program BASIC (H.C.-85)  
5 CLS
10. PRINT : PRINT "REPREZENTAREA GRAFICĂ A FUNCȚIEI:  $f(X) = X * X - |X|$ "
20. PRINT: PRINT "EXPLICITÎND MODULUL, AVEM:  
$$f(x) = \begin{cases} X * X - X, & X \geq 0 \\ X * X + X, & X < 0 \end{cases}$$
40. PRINT AT 21,0; "TASTAȚI ORICE": PAUSE 0
50. CLS : PLOT 30, 30 DRAW 200, 0 : DRAW -3,3 : DRAW 0, -6: DRAW 3,3
60. PLOT 130,9 :DRAW 0,165 :DRAW -3, -3 :DRAW 6,0 :DRAW -3,3:  
PRINT AT 0,17; "y"; AT 18,29; "X"; AT 19,15; "O"
70. FOR I=-3 TO 3 STEP 1/120
80. PLOT 130+30 \* I, 30+22 \* (I\*I-ABS I)
90. NEXT I
96. PRINT AT 19,7; "(-1,0)"; AT 19,19; "(1,0)"
100. PRINT AT 21,0; "FUNCȚIA DATA :  $(f(X) = X * X - |X|$ ": PAUSE 0  
: STOP

## 17.13. Program „ABC 2”

- 1°. Tema: „Obținerea tripletelor de numere pitagoreice”
- 2°. Program BASIC (HC-85)  
5 CLS
10. PRINT "PROGRAM DE OBTÎNERE A TRIPLETELOR DE NUMERE PITAGOREICE"
20. PRINT "INTRODUCEȚI NUMĂRUL N"
30. INPUT N
40. PRINT "N="; N
60. FOR R=2 TO N
80. FOR S=1 TO N
90. LET C=R \* R+S \* S
100. LET A=R \* R-S \* S
110. LET B=2 \* R \* S
120. IF C>N THEN GO TO 170
130. IF A<0 THEN GO TO 170
140. PRINT "A="; A
150. PRINT "B="; B

## 17. PROGRAME CLASELE I-VIII



```

160. PRINT "C=" ; C
170. IF S>=N THEN GO TO 200
180. LET S=S+1
190. NEXT S
200. LET R=R+1
210. IF R>N THEN GO TO 230
220. NEXT R
230. STOP

```

#### 17.14. Program „FUNCTIA”

1°. Temă: Subprogram de trasare a graficului funcției  $f$ ,  $f(X)=a * X * X$ ,  $X \in R$ , pentru coeficienți „a” negativi.

2°. Program BASIC (HC-85) – „FUNCTIA”

```

1. PRINT AT 1,2; „FUNCTIA”; PRINT AT 2,0; „f(X)=a * X * X”:
  PRINT AT 3,1; „GRAFICE (a<0)”
3. PLOT 10,170 : DRAW 235,0: DRAW -3,3: DRAW 3,-3: DRAW -3,
  -3:
  PLOT 125,0 : DRAW 0,175 : DRAW 3,-3 : DRAW -3,3 : DRAW -3,
  -3
20. FOR I=-40 TO 40 STEP 0.1
30. PLOT 125+1,170-0.08 * I * I
40. NEXT I
41. PRINT AT 16,21; „f(X)=-X * X”
50. FOR I=-40 TO 40 STEP 0.1
55. PLOT 125+I,170-0.08 * (1/2) * I * I
60. NEXT I
61. PRINT AT 7,24; „f(X)=-“ : PRINT AT 8,21; „=-(1/2) * X * X”
70. FOR I=-40 TO 40 STEP 0.1
72. PLOT 125+I,170-0.08 * (1/10) * I * I
80. NEXT I
81. PRINT AT 1,24; „f(X)=-“: PRINT AT 2,21; „-(1/10) * X * X”
90. FOR I=-34 TO 34 STEP 0.1
91. IF 170-0.08 * 2 * I * I<20 THEN GO TO 100
92. PLOT 125+I,170-0.08 * 2 * I * I
100 NEXT I
101. PRINT AT 18,20; „f(X)=-2 * X * X”
110 FOR I=-16 TO 16 STEP 0.05
111. IF 170-0.08 * 10 * I * I * 5 THEN GO TO 120
115. PLOT 125+170-0.08 * 10 * I * I
120. NEXT I
121. PRINT AT 21,18; „f(X)=-10 * X * X”

```



## 17.15. Program „Boyle Mariotte (I)”

Tema: Rezolvări de probleme (I) — legea Boyle-Mariotte.

```

1 REM Program "BOYLE-MARIOTTE"
2 REM 1'. Tema Rezolvări de probleme (I), legea Boyle-Mariotte
3 REM 2'. Program BASIC (HC-85)
4 REM 3'. Program BASIC (HC-85)

10 PRINT AT 2,0;" APLICATII BOYLE-MARIOTTE"
20 PRINT AT 6,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
30 PRINT AT 8,0;"La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
40 PRINT AT 10,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
50 PRINT AT 12,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
60 PRINT AT 14,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
70 PRINT AT 16,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
80 PRINT AT 18,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
90 PRINT AT 20,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
100 PRINT AT 22,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
110 PRINT AT 24,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
120 PRINT AT 26,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
130 INPUT "pi=";pi
140 IF pi<=0 THEN GO TO 180
150 PRINT "pi=";pi
160 INPUT "pf=";pf
170 IF pf<=0 THEN GO TO 180
180 PRINT "pf=";pf
190 INPUT "ui=";ui
200 IF ui<=0 THEN GO TO 240
210 PRINT "ui=";ui
220 INPUT "uf=";uf
230 IF uf<=0 THEN GO TO 240
240 PRINT "uf=";uf
250 INPUT "volumul initial este dat de:"
260 PRINT "for-- mola: ui=uf*pf/pi"
270 PRINT "for-- mola: ui=uf*pf/pi"
280 GO TO 280
290 CLS: PRINT "Introducerea datelor:"
300 PRINT " [U]=cmHg"
310 PRINT AT 3,21;"[p]=atm."
320 PRINT "Presiunea initiala:"
330 INPUT "pi=";pi
340 IF pi<=0 THEN GO TO 330
350 PRINT "pi=";pi
360 INPUT "pf=";pf
370 IF pf<=0 THEN GO TO 330
380 PRINT "pf=";pf
390 INPUT "ui=";ui
400 IF ui<=0 THEN GO TO 330
410 PRINT "ui=";ui
420 INPUT "uf=";uf
430 IF uf<=0 THEN GO TO 330
440 PRINT "uf=";uf
450 PRINT "for-- mola: ui=uf*pf/pi"
460 PRINT "for-- mola: ui=uf*pf/pi"
470 GO TO 470
480 CLS: PRINT "DACA DORITI SA CONTINUATI TASTATI A; DACA NU, TASTATI ORICE"
490 PAUSE 0: CLS
500 IF INKEYS="A" THEN GO TO 120
510 STOP

```

## 17.16. Program „Boyle Mariotte (II)”

Tema: Rezolvări de probleme (II). Legea Boyle-Mariotte

```

1 REM Program "BOYLE-MARIOTTE"
2 REM 1'. Tema Rezolvări de probleme (II), legea Boyle-Mariotte
3 REM 2'. Program BASIC (HC-85)

10 PRINT AT 2,0;" APLICATII BOYLE-MARIOTTE"
20 PRINT AT 6,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
30 PRINT AT 8,0;"La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
40 PRINT AT 10,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
50 PRINT AT 12,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
60 PRINT AT 14,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
70 PRINT AT 16,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
80 PRINT AT 18,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
90 PRINT AT 20,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
100 PRINT AT 22,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
110 PRINT AT 24,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
120 PRINT AT 26,0;"Se da problema: La presiunea de a cmHg, un gaz ocupa un volum de b cmHg. Cea de a cmHg, un gaz ocupa un volum de c cmHg. Se cere presiunea finala a gazului."
130 INPUT "pi=";pi
140 IF pi<=0 THEN GO TO 180
150 PRINT "pi=";pi
160 INPUT "pf=";pf
170 IF pf<=0 THEN GO TO 180
180 PRINT "pf=";pf
190 INPUT "ui=";ui
200 IF ui<=0 THEN GO TO 240
210 PRINT "ui=";ui
220 INPUT "uf=";uf
230 IF uf<=0 THEN GO TO 240
240 PRINT "uf=";uf
250 INPUT "volumul initial este dat de:"
260 PRINT "for-- mola: ui=uf*pf/pi"
270 PRINT "for-- mola: ui=uf*pf/pi"
280 GO TO 280
290 CLS: PRINT "Introducerea datelor:"
300 PRINT " [U]=cmHg"
310 PRINT AT 3,21;"[p]=atm."
320 PRINT "Presiunea initiala:"
330 INPUT "pi=";pi
340 IF pi<=0 THEN GO TO 330
350 PRINT "pi=";pi
360 INPUT "pf=";pf
370 IF pf<=0 THEN GO TO 330
380 PRINT "pf=";pf
390 INPUT "ui=";ui
400 IF ui<=0 THEN GO TO 330
410 PRINT "ui=";ui
420 INPUT "uf=";uf
430 IF uf<=0 THEN GO TO 330
440 PRINT "uf=";uf
450 PRINT "for-- mola: ui=uf*pf/pi"
460 PRINT "for-- mola: ui=uf*pf/pi"
470 GO TO 470
480 CLS: PRINT "DACA DORITI SA CONTINUATI TASTATI A; DACA NU, TASTATI ORICE"
490 PAUSE 0: CLS
500 IF INKEYS="A" THEN GO TO 120
510 STOP

```



**Tema: Program „FRE CARE“**

```

10>CLS
20>PRINT "CONCENTRATIA PROCENT
30>PRINT "MASA SUBSTANTEI DIZU
40>PRINT "IN GRAME"
50>INPUT md
60>IF md<=0 THEN GO TO 40
70>PRINT "md="";md
80>PRINT "MASA SOLUTIEI, IN GR
90>INPUT ms
100>IF ms<=0 THEN GO TO 80
110>PRINT "ms="";ms
120>PRINT "CONCENTRATIA PROCENT
130>PRINT "c="";c
140>STOP

```



## 17.19. Program „NUMERE POLINDROAME” (cerc de matematică)

### Tema “Numere polindroame”

```

1  CLS : GO SUB 2000
2  INPUT "Nr Bazelor (<=4) :";
b: IF b>0 THEN DIM b(b): FOR i=1
to b: INPUT "Baza :";i;"=");b
(i): NEXT i
10 INPUT "Introduceti numarul
n=";n: LET qq=1: LET j=1: DIM a(
1000)
15 IF n<=9 THEN FOR i=1 TO n:
LET a(i)=i: RANDOMIZE USR 3582:
PRINT AT 20,2;"a(i)="+a(i): PAUSE
20: RANDOMIZE USR 3582: PRINT AT 20,2
;"a(i)="+a(i): PAUSE 20: RANDOMI
ZE USR 3582: PRINT : NEXT i
30 LET k=10
40 FOR i=1 TO 9
50 LET a(k)=11*i
55 IF a(k)=n THEN GO TO 301
60 LET k=k+1: RANDOMIZE USR 35
82: PRINT AT 20,2;"a(k-1)="+a(k-1):
-1: RANDOMIZE USR 3582: PRINT
70 NEXT i
100 FOR w=10+(j-1) TO 10+j-1
110 FOR i=0 TO 9
120 LET b$=STR$ i
130 LET b$=STR$ w
132 DIM c$(LEN b$)
135 FOR t=1 TO LEN b$: LET c$(L
EN b$-(t+1))=b$(t): NEXT t
140 LET d$=b$+a$+c$
150 LET m=VAL d$
155 IF m>n THEN GO TO 301
170 LET a(k)=m: LET k=k+1
180 RANDOMIZE USR 3582: PRINT A
T 20,2;"a(k-1)="+a(k-1):
ZE USR 3582: PRINT
185 PAUSE 10
190 NEXT i: NEXT w
200 FOR w=10+(j-1) TO 10+j-1
210 FOR i=0 TO 99 STEP 11
220 LET a$=STR$ i
225 IF i=0 THEN LET a$="00"
230 LET b$=STR$ w
232 DIM c$(LEN b$)
235 FOR t=1 TO LEN b$: LET c$(L
EN b$-(t+1))=b$(t): NEXT t
240 LET d$=b$+a$+c$
250 LET m=VAL d$
255 IF m>n THEN GO TO 301
270 LET a(k)=m: LET k=k+1
280 RANDOMIZE USR 3582: PRINT A
T 20,2;"a(k-1)="+a(k-1):
ZE USR 3582: PRINT
285 PAUSE 10
290 NEXT i: NEXT w
300 LET j=j+1: GO TO 100
301 IF b=0 THEN STOP
302 PRINT #1; FLASH 1;"TASTATI
ORICE PENTRU A CONTINUA!"; PAUSE
0: CLS: LET p=1: PRINT "Bazele sint
": PRINT
304 PRINT TAB 0;"10";
305 FOR u=1 TO b: PRINT TAB 31
(b+1)*u-4;b(u): NEXT u: DIM R(4
0,b): DIM c(k-1,b): DIM v$(k,b,12)
305 FOR i=0 TO 31: PRINT TAB i;
"10"; NEXT i
307 PRINT
308 FOR u=1 TO k-1
309 LET f=a(y)
310 FOR j=1 TO b
320 LET t=1
325 LET a(y)=f
330 LET a(t,j)=a(y)-b(j)*INT (a
(y)/b(j))
335 LET a(y)=INT (a(y)/b(j))
340 IF a(y)=0 THEN GO TO 360
350 LET t=t+1: GO TO 330
360 FOR u=1 TO INT (t/2)
370 IF a(u,j)<>a(t+1-u,j) THEN
GO TO 410
380 NEXT u
385 FOR u=1 TO t: LET v$(p,j,t+
1-u)=STR$ a(u,j): NEXT u
390 NEXT j
395 LET p=p+1:
400 PRINT f
405 IF u=1 TO b-1
404 IF u=1 THEN PRINT TAB 31/(b
+1)-4;"10"; PRINT TAB 31/b-4;"v$(
p-1,
405 IF u=2 THEN PRINT TAB 31/(
b+1)*u-7;"10"; PRINT TAB 31/b*u-
4;"v$(p-1,u)";
406 NEXT u
407 PRINT TAB 31*b/(b+1)-7;"10";
PRINT TAB 31*b/(b+1)-4;"v$(p-1,
b)";
410 NEXT y
420 STOP
2000 PRINT AT 10,10; FLASH 1;"ST
OP THE TAPE"
2010 PAUSE 50: CLS
2015 PRINT AT 0,4; PAPER 2; INK
6;"NUMERE PALINDROAME": PRINT
2020 LET l$=
Accest program calculeaza toate
numerele PALINDROAME mai mici
decit un numar "n" dat.
Prin numar palindrom intelegem
acel numar care are aceeasi va-
loare fie ca-l citim dintr-un
sens, fie din sens opus."
2025 FOR i=1 TO LEN l$: PRINT IN
K 1;l$(i): BEEP .05,30: NEXT i
2030 PRINT : PRINT "EXEM
PLU:"; PRINT FLASH 1;" n=21312"
2040 PRINT
2050 PAUSE 50
2060 PRINT
Programul este structurat pe
doua idei:
i) Numerele sint calculate in
baza zece,
ii) Numerele sint trecute in al
te baze de numeratie si apoi
sint verificate daca sint
numere palindroame"
2070 PRINT #1; FLASH 1;"TASTATI
ORICE PENTRU A CONTINUA!";
2080 PAUSE 0
2090: CLS
2100 PRINT AT 5,0;
In cazul in care doriti nume-
re palindroame doar in baza
zece, introduceti ca numar de
baze "0", apoi introduceti nu-
marul.
In cazul ii) introduceti nu-
marul de baze si apoi bazele"
2110 PRINT #1; FLASH 1;"TASTATI
ORICE PENTRU A CONTINUA!";
2120 PAUSE 0
2130 RETURN

```



## 17.20. Program „RELAȚII”

1°. Tema: Subprogram pentru trasarea graficului funcției f,

$$f(X) = \begin{cases} -X-4, & X \leq -2 \\ X, & -2 < X \leq -1 \\ -\frac{1}{4}X - \frac{5}{4}, & -1 < X \leq -3 \\ X-5, & X > 3 \end{cases}$$

și a relației  $|Y| = |f(X)|$

(Cerc de matematică)

2°. Program BASIC (HC-85)—„RELAȚII”

```

5990. GO TO 7000
6000. CLS : PRINT "FIE GRAFICUL LUI Y=f(X)
6010. PLOT 5,50 : DRAW "245,0 : PLOT 125,5 : DRAW 0,165 : PRINT
      AT 4,16; "Y"; AT 15,31; "X"
6020. PRINT AT 16,16; "0": PRINT AT 16,8; "-3": PRINT AT 16,2; "-5":
      PRINT AT 16,0; "6"
6030. FOR X=1 TO 245 STEP 2
6040. IF X<80 THEN LET Y=90-X
6050. IF X>=80 AND X<100 THEN LET Y=X-70
6060. IF X>=100 AND X<180 THEN LET Y=-0.25 * X+55
6065. IF X>=180 THEN LET Y=X-170
6070. IF X=81 THEN PRINT AT 21,7; "(-2, -2)"
6080. IF X=41 THEN PRINT AT 16,4; "-4"
6090. IF X=181 THEN PRINT AT 21,20; "(3, -2)"
6100. IF X=101 THEN PRINT AT 16,12; "-1": PRINT AT 17,16; "-1"
6110. PLOT X+5, Y
6120. IF X=221 THEN PRINT AT 16,28; "5.6"
6130. NEXT X
6140. RETURN
7000. CLS : GO SUB 6000 : FOR X=1 TO 245
7005 PRINT AT 1,0; "REALIZĂM GRAFICUL RELAȚIEI |Y|=|f(X)| notat
      R"
7010. IF X<80 THEN LET Y=90-X
7020. IF X>=80 AND X<100 THEN LET Y=X-70
7030. IF X>=100 AND X<180 THEN LET Y=0.25 * X+55
7040. IF X>=180 THEN LET Y=X-170
7050. PLOT X+5,52+ABS (Y-50): PLOT X+5,48-ABS (Y-50)
7060. NEXT X
7065. PRINT AT 9,10; "R"
7070. STOP

```



### 17.21. Program „ROT. TRIUNGHI”

1°. Tema: Rotirea unui triunghi oarecare în jurul laturii considerată ca bază.

2°. Program BASIC (HC-85) — „ROT. TRIUNGHI”

```
10. FOR I=0 TO 4 * PI STEP 0.5
20. PLOT 100,80 : DRAW 145,0
30. DRAW -20-(20-20 * COS I), 70 * SIN I
40. PLOT 100,80 : DRAW 105+20 * COS I, 70 * SIN I
50. NEXT I
60. FOR I=0 TO 4 * PI STEP 0.03
70. PLOT 205+20 * COS I, 80+70 * SIN I
90. NEXT I
100. STOP
```

### 17.22. Program „ROT. TRAPEZ”

1°. Tema: Rotirea unui trapez în jurul bazei mari.

2°. Program BASIC (HC-85) — „ROT. TRAPEZ”

```
10. FOR I=0 TO 4 * PI STEP 0.5
20. PLOT 60, 60 : DRAW 150,0
21. PLOT 70+(20-10 * COS I), 60+40 * SIN I : DRAW 80,0
40. PLOT 60,60 : DRAW 10+(20-10 * COS I), 40 * SIN I : PLOT 210,
    60 : DRAW -40 -10 * COS I, 40 * SIN I
50. NEXT I
60. FOR I=0 TO 4 * PI STEP 0.03
70. PLOT 70+20-10 * COS I, 60+40 * SIN I
80. PLOT 210-40-10 * COS I, 60+40 * SIN I
90. NEXT I
100. STOP
```

### 17.23. Program „ROT. HEXAGON”

1°. Tema: Rotirea unui hexagon în jurul unei laturi.

2°. Program BASIC (HC-85) — „ROT. HEXAGON”

```
10. PLOT 125,60 : DRAW 0,40 : DRAW -20 * SQR 3,20 : DRAW -20
    SQR 3, -20 : DRAW 0, -40 : DRAW 20 * SQR 3, -20 : DRAW 20 *
    SQR 3,20
20. FOR I=PI TO 5 * PI STEP 0.5
30. PLOT 125,100 : DRAW 20 * SQR 3 * COS I, 20 -5 * SIN I
40. DRAW 40 * SQR 3 * COS I -20 * SQR 3 * COS I, -(20+5 * SIN
    I+10 * SIN I)
```



```

50. DRAW 0, -40
60. PLOT 125,60: DRAW 20 * SQR 3 * COS I, -20-5 * SIN I
70. DRAW 40 * SQR 3 * COS I -20 * SQR 3 * COS I, 20-5 * SIN I
   -10 * SIN I
80. NEXT I
90. FOR I=PI TO 3 *PI STEP 0.04
100. PLOT 125+20 * SQR 3 * COS I, 120+5 * SIN I
110. PLOT 125+40 * SQR 3 * COS I, 100+20 * SIN I
120. PLOT 125+20 * SQR 3 * COS I, 40+5 * SIN I
130. PLOT 125+40 * SQR 3 * COS I, 60+20 * SIN I
140. NEXT I: STOP

```

## 17.24. Program „POLINOM”

1°. TEMA: Calculul valorii unui polinom de gradul N.

2°. Program BASIC (H.C.-85)

```

10. CLS
20. PRINT "CALCULUL VALORII UNUI POLINOM DE GRADUL N"
30. PRINT "P(X)=A0 * XN+A1 * X(N-1)+...+AN"
40. PRINT "INTRODUCETI GRADUL POLINOMULUI :N"
50. INPUT N
60. PRINT "N="; N
70. PRINT "INTRODUCETI COEFICIENTII"
80. DIM A(N+1)
90. FOR I=1 TO N+1
100. PRINT "A("; I-1; ")=";
110. INPUT A(I):PRINT A(I)
120. NEXT I
130. PRINT "X=?"
140. INPUT X
150. PRINT "X="; X
160. LET P= *
170. FOR K= * TO N
180. LET C=N-K
181. IF 2 * INT(C/2)=C THEN LET P=P+A(K+1) * (ABS X)↑ C
182. IF 2 * INT(C/2)<>C THEN LET P=P+A(K+1) * SGN x * (ABS x)↑ C
190. NEXT K
200. PRINT "VALOAREA POLINOMULUI ESTE"
210. PRINT "P (";X;")=";P
220. STOP

```



## 17.25. Program „TRIUNGHI”

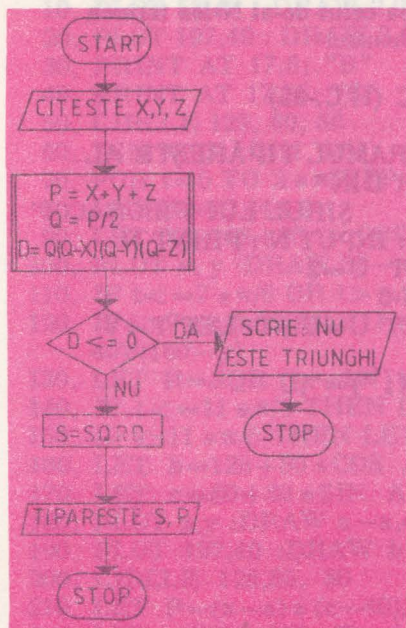


Fig. 17.10. Schema logică „Triunghi”

- 1°. Tema: Program de calculare a perimetrului și ariei unui triunghi cînd se cunosc laturile: X, Y, Z.
- 2°. Schemă logică
- 3°. Program BASIC (HC-85)—„PERIMETRU”
10. PRINT „CALCULUL PERIMETRULUI ȘI CALCULUL SUPRAFEȚEI TRIUNGHIULUI”
20. PRINT ”SE CUNOSC LUNGIMILE LATURILOR”
30. PRINT ”DAȚI LUNGIMILE LATURILOR”
40. INPUT X, Y, Z
50. LET P=X+Y+Z
60. LET Q=P/2
70. LET D=Q\*(Q-X)\*(Q-Y)\*(Q-Z)
80. IF D=0 THEN PRINT ”NU ESTE TRIUNGHI” : GO TO 130
90. LET S=SQRT D
100. PRINT ”PERIMETRUL :”; P
110. PRINT ”ARIA SUPRAFEȚEI :”; S
130. STOP

Programul „TRIUNGHI” a fost mult dezvoltat în cadrul cercului de matematică de la liceul D. Cantemir, datorită colaborării cu prof. Radu Jugureanu și elevul Maxim Iurie. Recomandăm, să se realizeze programe de tip expert-conversațional, care pot cuprinde următoarele idei:

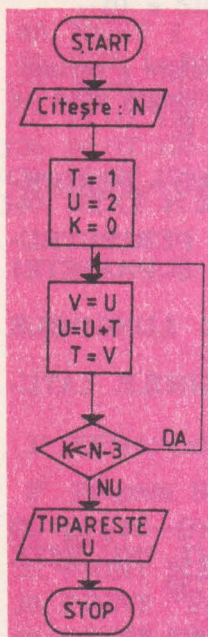
1. Mulțimea datelor de intrare să poată fi selectiv (și în același timp interactiv) folosită; spre exemplu, programul amintit are 13 date de intrare (laturi, unghiuri, mediane, perimetru, arie, raza cercului înscris și a celui circumscris), apelabile interactiv în funcție de problema dată.
2. Mulțimea datelor de ieșire să fie, de asemenea, opțional utilizată.

Programul „TRIUNGHI” are posibilitatea de a rezolva 286 de cazuri distincte de probleme de construcție, pentru fiecare caz calculînd valorile a 19 elemente ale triunghiului respectiv.

3. Posibilitatea de a compune alte probleme pe scheletul celor formulate inițial; programul sus-menționat poate crea pentru fiecare din cele 286 de cazuri distincte 1023 probleme noi, prin combinarea diferitelor elemente ale triunghiului rezultînd un număr de 292578 probleme distincte.



## 17.26. Program „FIBONNACI”



1°. Tema: Calcularea celui de-al N-lea termen din șirul lui "Fibonacci"  
 2°. Schema bloc.  
 3°. Program BASIC (H.C.-85)  
 5 CLS  
 10. PRINT "PROGRAMUL TIPĂREȘTE AL N-LEA NUMĂR DIN"  
 20. PRINT "ȘIRUL LUI FIBONACCI"  
 30. PRINT "N="; : INPUT N : PRINT N  
 50. LET T=1: LET U=2  
 70. FOR K=0 TO N-3  
 80. LET V=U : LET U=U+T: LET T=V  
 100. NEXT K  
 110. PRINT "F ("; N; ")="; U  
 120. STOP

Fig. 17.11. Schema logică „Fibonacci”

### Observații:

„Șirul”  $a_n$ ,  $n$  număr natural,  $n_1 \geq 1$ , definit prin relațiile:  $a_1 = a_2 = 1$ ,  $a_{n+3} = a_n + a_{n+1}$  este numit șirul lui FIBONACCI.

Dăm, spre exemplificare, primii douăzeci de termeni ai acestui șir:  $a_1 = a_2 = 1$ ,  $a_3 = 2$ ,  $a_4 = 3$ ,  $a_5 = 5$ ,  $a_6 = 8$ ,  $a_7 = 13$ ,  $a_8 = 21$ ,  $a_9 = 34$ ,  $a_{10} = 55$ ,  $a_{11} = 89$ ,  $a_{12} = 144$ ,  $a_{13} = 233$ ,  $a_{14} = 377$ ,  $a_{15} = 610$ ,  $a_{16} = 987$ ,  $a_{17} = 1597$ ,  $a_{18} = 2584$ ,  $a_{19} = 4181$ ,  $a_{20} = 6765$ .

S-a demonstrat că pentru  $n \in \{3, 4, 5, 7, 11, 13, 17, 23, 29, 43, 47\}$  se obțin numere  $a_n$  prime. Alte numere primul  $a_n$ , de tipul FIBONACCI, încă nu se cunosc. Nu se știe dacă așa-numitul șir al lui FIBONACCI conține o infinitate de numere prime.

Se demonstrează că dacă  $n \neq 4$  și numărul  $a_n$  este prim, atunci  $n$  este prim. Reciproca nu este adevărată, deoarece avem:  $a = 1$ ,  $a = 113 \cdot 47, \dots$ . Nu se cunoaște nici faptul dacă în mulțimea numerelor  $a_n$ ,  $n$  prim, există o infinitate de numere compuse.



## 17.27. Subprogram „TRASARE LOC GEOMETRIC“

1°. Problemă Se cere locul geometric al centrului cercului înscris într-un triunghi, atunci când două vîrfuri sînt fixe, iar al treilea se mișcă pe cercul circumscris triunghiului.

2°. Program BASIC — „TRASARE LOC GEOMETRIC“

```

10. PLOT 58,40 : DRAW 2,0 : DRAW 0,2 : DRAW -2,0 : DRAW 0,-2
20. PLOT 197,40 : DRAW 2,0 : DRAW 0,2 : DRAW -2,0 : DRAW 0,-2
30. PRINT AT 17,6; "B"
40. PRINT AT 17,25; "C"
50. CIRCLE 128, 80, 80
60. DIM X(63) : DIM Y(63)
70. FOR t=0 TO 2 *  $\pi$  STEP 0,1
80. LET K=10 * t + 1
90. LET X=128+80 * COS t: LET Y=80+80 * SIN t
100. PLOT x, y : DRAW 58-x,40-y: DRAW 139,0 : DRAW x-197,y-40
110. IF t <= 7 *  $\pi$ /6 OR t >= 11 *  $\pi$ /6 THEN LET A=128+40 * (x-128)/y
120. IF t > 7 *  $\pi$ /6 AND t < 11 *  $\pi$ /6 THEN LET A = (120 * x - 128 * y + 128 * 40)/(160-y)
130. LET R=SQR ((x-197) * (x-197) + (y-40) * (y-40))
140. IF t <= 11 *  $\pi$ /6 THEN LET ALFA=7 *  $\pi$ /12 + t/2
150. IF t > 11 *  $\pi$ /6 THEN LET ALFA = 19 *  $\pi$ /12 - t/2
160. LET b=128+80 * COS ALFA
170. LET c=80+80 * SIN ALFA
180. PLOT x,y : DRAW a-x,40-y
190. PLOT 197,40 : DRAW b-197, c-40
200. CIRCLE 128,80, 80
210. LET P=(x-a) * (c-40)/((y-40) * (b-197))
220. LET x(K)=(197 * P - a)/(P-1)
230. LET y(K)=40+(x(K)-a) * (y-40)/(x-a)
240. FOR I=1 TO K: PLOT x(I), y(I): NEXT I
250. PLOT x, y : DRAW INVERSE 1; a-x, 40-y
260. PLOT 197,40 : DRAW INVERSE 1; b-197, c-40
270. PLOT x,y : DRAW INVERSE 1: 58-x,40-y
280. PLOT 197,40 : DRAW INVERSE 1; x-197, y-40
290. NEXT t
300. PAUSE 500 : CLS
310. FOR I=1 TO 63 : PLOT x(I), y(I): NEXT I;
320. STOP

```

## 17.28. Subprogram „DESENAREA UNEI TABLE DE ȘAH“

```

5. CLS
10. PRINT "PROGRAMUL TIPĂREȘTE O TABLĂ DE ȘAH": PAUSE 300:
LET A=10 GO SUB 150:
20. FOR J=1 TO 8
30. FOR I=A TO 6+A STEP 2
40. PRINT AT 16-J,I; INK C; " "
50. NEXT I
60. IF A=10 THEN LET A=11; GO TO 80
70. IF A=11 THEN LET A=10
80. NEXT J

```



```

90. PRINT AT 16,10; "ABCDEFGH"
100. PRINT AT 7,10; "ABCDEFGH"
110. FOR I=1 TO 8 : PRINT AT 16 -I,9; I:NEXT I
120. FOR I=1 TO 8 : PRINT AT 16 -I, 18; I: NEXT I
130. STOP
150. INPUT "INTRODUCETI : CULOAREA CAROURILOR DE NUANTĂ
    ÎNCHISĂ (0-7): "; C : RETURN

```

## 17.29. „GRAFICE DE FUNCȚII”

1°. Tema: Subprogram pentru trasarea graficului funcției f,

$$f(X) = \begin{cases} -X-4, & X \leq -2 \\ X, & -2 < X \leq -1 \\ -\frac{1}{4}X - \frac{5}{4}, & -1 < X \leq 3 \\ X-5, & X > 3 \end{cases}$$

și a modului acestei funcții.

(Cerc de matematică)

2°. Program BASIC (HC-85) – „GRAFICE DE FUNCȚII”

```

740. CLS
750. PLOT 5,50 : DRAW 245,0 : PLOT 125,5 : DRAW 0,165 : PRINT AT
    4,16; "y"; AT 15,31; "x"
760. PRINT AT 16,16; "0" : PRINT AT 16,8; "-3" : PRINT AT 16,2; "-5"
    PRINT AT 16,0; "-6"
780. FOR X=1 TO 245 STEP 2
790. IF X<80 THEN LET Y=90-X
800. IF X>=80 AND X<100 THEN LET Y=X-70
810. IF X>=100 AND X<180 THEN LET Y=-0.25 * X+55
820. IF X>=180 THEN LET Y=X-170
823. IF X=81 THEN PRINT AT 21,7; "(-2, -2)"
826. IF X=41 THEN PRINT AT 16,4; "-4"
827. IF X=101 THEN PRINT AT 16,12; "-1" : PRINT AT 17,16; "-1"
829. IF X=181 THEN PRINT AT 21,20; "(3, -2)"
830. PLOT X+5, Y
835. IF X=221 THEN PRINT AT 16,28; "5 6"
840. NEXT X
860. FOR X=1 TO 245
870. IF X<40 THEN LET Y=90-X
880. IF X>40 AND X<80 THEN LET Y=X+10
890. IF X>=80 AND X<100 THEN LET Y=170-X
900. IF X>=100 AND X<180 THEN LET Y=0.25 * X+45
910. IF X>=180 AND X<221 THEN LET Y=-X+270
920. IF X>=221 THEN LET Y=X-171
925. IF X=80 THEN PRINT AT 9,8; "(-2,2)"
927. IF X=100 THEN PRINT AT 13,16; "1"
929. IF X=181 THEN PRINT AT 9,21; "(3,2)"
930. PLOT X+4, Y+3
940. NEXT X
950. STOP

```

\* Programul 17.30 este la pag. 119 jos.



## Capitolul 18. | Programe pentru clasele IX-XII

Programele și subprogramele care fac obiectul acestui capitol au fost experimentate și validate în cadrul Liceului Industrial „Dimitrie Cantemir” la CERCUL EXPERIMENTAL\*) PENTRU FOLOSIREA INFORMATICII ȘI CALCULATORULUI ÎN PROCESUL DE ÎNVĂȚĂMÎNT.

La testarea și experimentarea programelor au participat elevi și cadre didactice din liceu\*. Programele informatice educaționale aparțin unor teme din programele școlare de matematică, de fizică, de chimie și de electrotehnică... Ele „încearcă” să răspundă exigențelor privind folosirea calculatorului pentru ilustrarea unor momente sau secvențe ale lecției — în procesul de predare, pentru munca de pregătire individuală sau în grup — cu ajutorul calculatorului, pentru optimizarea unor teme adresate cercurilor de elevi precum și pentru munca de cercetare în care se folosește calculatorul. Evident ele sînt perfectibile.

Așa cum s-a mai afirmat în cuprinsul acestei lucrări, informatica și calculatorul se integrează astăzi treptat, dar generalizat, procesului de învățămînt, atît ca obiect de învățămînt, cît și ca mijloc de învățămînt și de producție. Informatica și calculatorul constituie azi o sursă de teme atît pentru licee, cît și pentru cercurile pionierești și taberele de vacanță, atît pentru studenți, cît și pentru profesioniști.

Așa cum se știe, folosirea calculatorului ca mijloc de învățămînt conferă procesului învățării un caracter interactiv, bazat pe psihologia acțiunii. Utilizat corect, el permite prezentarea cu animație, în procesul de predare-învățare, a unei largi varietăți de scheme, machete, diagrame, desene și modele intuitive (fie simplificate, fie luate direct din realitatea practică). Calculatorul permite simularea, reproducind fenomene și procese în mișcare, asupra cărora își concentrează atenția cel care trebuie să învețe, sau să cerceteze. Modelele, experimentul și animația permit o reprezentare dinamică a spațiului și nu statică. Aspectele dinamice din prezentarea locurilor geometrice, a transformărilor geometrice și a geometriei vectoriale dezvoltă la elevi capacitatea de a vizualiza în mișcare imaginile. Spre exemplu, un program care vizualizează pe ecran locul geometric al ortocentrului unui triunghi ABC, A și B fixe, iar C mobil pe cercul circumscris triunghiului ABC permite intuirea locului căutat, facilitînd apariția ideilor necesare demonstrației. Odată intuit acest loc geometric și realizată demonstrația (sintetic sau analitic), elevii pot prelungi modelul anterior investigînd modalități de realizare grafică a locurilor geometrice ale intersecției medianelor sau bisectoarelor, punctul C fiind mobil pe cercul circumscris. Se poate trece, în continuare, la căutarea de locuri geometrice obținute prin intersecții de drepte, de cercuri sau alte curbe — cu respectarea apunîtor condiții. Vizuali-



zarea graficelor unor funcții permite observarea comportării acestora în vecinătatea unor puncte caracteristice; se pot face o serie de legături cu proprietățile de continuitate și derivabilitate; se pot rezolva grafic ecuații de forma  $f(x)=g(x)$ .

De asemenea, calculatorul face posibilă aplicarea unor algoritmi de prelucrare rapidă a datelor, solicită participarea activă și inițiativă în explorarea necunoscutului, investigare și verificare experimentală, interpretare teoretică. În acest proces este solicitată intuiția și sînt folosite intensiv măsurarea, compararea, ordonarea, conjugarea și transferul, clasificarea și generalizarea. Pe această bază se dezvoltă spiritul programatic, creativitatea și descoperirea dirijată.

În cele ce urmează sînt prezentate secvențe de programe, subprograme și cîteva programe complete pentru învățămîntul liceal, îndeosebi pentru clasele a IX-a și a X-a\*\*.

Ele pot fi luate, la începutul activității, ca ghid pentru folosirea calculatorului în timpul unei ore, în deosebi în clasele a IX-a și a X-a. Sub îndrumarea profesorului, elevii pot realiza schema logică, în baza căreia vor trece apoi la scrierea programului în BASIC. După aceasta se tastează programul, se verifică și se salvează — dacă este cazul. Evident după o oarecare experiență, elevii se pot desprinde de scrierea schemei logice, trecînd direct la scrierea programului în BASIC. În acest mod, elevii pot păși în etapa muncii lor creatoare pentru realizarea de noi programe originale, în cadrul diverselor discipline școlare.

---

\* Cercul amintit constituie rodul colaborării Liceului Industrial „Dimitrie Cantemir” — București, cu Facultatea de Automatică a I.P.B. prin prof. univ. dr. Adrian Petrescu. În acest sens există un protocol în baza căruia se efectuează experimentul privind procesul de „informa-tizare a învățămîntului și de testare a calculatoarelor didactice”.

În realizarea experimentului, privind scrierea și folosirea la clasă a unor programe, și-au adus contribuția o serie de cadre didactice din liceu: Jugureanu Radu, Crăciun Marin, Văgîli Ștefan, Demșorean Adrian, Demșorean Anca, Coteș Mariana, Iarca Marina și Scarlat Aurelia. De asemenea, au participat la scrierea și experimentarea programelor o serie de elevi din echipa de informatizare a liceului: Petrescu Iacob, Bălan Radu, Tudor Florin, Ionescu Adrian, Neghiu Adina, Rusu Iulia, Solomon Sorin, Suciu Andrei. Menționăm aici și sprijinul laborantului școlii, student Vicoveanu Gabriel.

\*\* Unele secvențe nu sînt încheiate, ele fiind susceptibile completării ulterioare de către utilizatori.



## 18.1. Program „FUNCȚIA MODUL”

1°. Tema: Subprogram pentru trasarea graficului funcției

$$f(x) = ||x+1| - 1|, x \in \mathbb{R}.$$

2°. Program BASIC (HC-85) – „FUNCȚIA MODUL”  
(Cerc de matematică)

- 1) TRASAREA DIRECTĂ
- 2) TRASAREA FOLOSIND TRANSLAȚIA ȘI SIMETRIA

```
180 CLS
190 PRINT FLASH 1; "1 GRAFICUL FUNCȚIEI f(X)=||X+1|-1|"
200 PRINT "1.1 MAI ÎNTÎI TRASAREA DIRECTĂ"
210 PLOT 5,30 : DRAW 245,0 : PLOT 140,0 : DRAW 0,120
215 PRINT AT 4,18; "y": PRINT AT 17,30; "x"
220 FOR I=5 TO 250
230 IF I<=100 THEN LET Y=130-I
240 IF I<=120 AND I>100 THEN LET Y=I-70
250 IF I<=140 AND I>120 THEN LET Y=170-I
260 IF I>140 THEN LET Y=I-110
265 IF I=120 THEN PRINT AT 14,12; "(-1,1)"
270 PLOT I, Y
280 NEXT I
290 PRINT AT 19,11; "-2,0"
300 PRINT AT 20,4; FLASH 1; "PENTRU A CONTINUA"
310 PRINT AT 21,8; FLASH 1; "APĂSAȚI ORICE TASTĂ"
320 INPUT INKEY
330 IF INKEY $=" " THEN GO TO 320
340 CLS
350 PRINT "1.2 FOLOSIREA PROCEDEULUI TRANSLAȚIEI PENTRU
    TRASAREA PE ETAPE A GRAFICULUI"
360 PRINT "i) SE REALIZEAZĂ GRAFICUL FUNCȚIEI f(X)=|X|"
370 PLOT 5,30 : DRAW 245,0 : PLOT 140,0 : DRAW 0,120
375 PRINT AT 4,18; "y": PRINT 17,30; "X": PAUSE 50
380 FOR I=45 TO 235 STEP 2
390 LET Y=30+ABS(I-140)
400 PLOT I, Y
410 NEXT I
415 PRINT AT 19,14; "-1"
419 PAUSE 200
420 PRINT AT 5,0; "ii) SE TRANSLATEAZĂ ACEST GRAFIC PE AXA OX
    CU -1"
430 FOR I=50 TO 232
440 LET Y=30+ABS(I-140)
450 PLOT I-20, Y
460 NEXT I
470 PAUSE 450 : CLS
```



```

480 PLOT 5,30 : DRAW 245,0 : PLOT 140,20 : DRAW 0,120
485 PRINT AT 4,18; "y" : PRINT AT 17,30; "x"
490 FOR I=30 TO 250 STEP 2
500 LET Y=ABS (I-140)
510 PLOT I-20, Y
520 NEXT I
530 PRINT AT 2,0; "iii) SE TRANSLATEAZĂ ACUM GRAFICUL OBTINUT
    PE AXA (oy) CU -1"
540 FOR I=30 TO 250
    HC-85
550 LET Y=30+ABS (I-140)
560 PLOT I-20, Y-20
565 IF I=140 THEN PRINT AT 21, 11; "(-1, -1)"
570 NEXT I : PAUSE 250 : CLS
571 PLOT 5,30 : DRAW 245,0 : PLOT 140,20 : DRAW 0,120
572 PRINT AT 5,18; "y" : PRINT AT 17,30; "x"
573 FOR I=25 TO 270 STEP 2
574 LET Y=30+ABS (I-140)
576 PLOT I-20, Y-20
577 IF Y=141 THEN PRINT AT 21,11; "(-1, -1)"
578 NEXT I
580 PRINT AT 0,0; „iv) PENTRU TRASAREA GRAFICULUI FUNCȚIEI
    CERUTE SE RIDICĂ PRIN SIMETRIA ÎN RAPORT CU AXA ox,
    PORȚIUNEA NEGATIVĂ A GRAFICULUI FUNCȚIEI DATE"
585 PRINT AT 5,18; "y" : PRINT AT 17,30; "x"
590 FOR I=5 TO 250
600 IF I<=100 THEN LET Y=130-I
610 IF I<=120 AND I>100 THEN LET Y=I-70
620 IF I<=140 AND I>120 THEN LET Y=170-I
630 IF I>140 THEN LET Y=I-110
635 IF I=120 THEN PRINT AT 14,11; "(-1,1)"
640 PLOT I, Y+2
650 NEXT I
655 PAUSE 0

```

OBSERVAȚIE: INSTRUCȚIUNILE: 210 și 215

370 și 375

480 și 485

571 și 572

pot fi înlocuite cu instrucțiunile:

210 GO SUB 1000

370 GO SUB 1000

480 GO SUB 1000

571 GO SUB 1000

1000 PLOT 5,30 : DRAW 245,0 :  
PLOT 140,0 : DRAW 0,120

1010 PRINT AT 4,18; "y" : PRINT  
AT 17,30; "x"

1020 RETURN

folosindu-se astfel noțiunea de sub-  
rutină.



## 18.2. Program „FUNCȚIA”

1°. Temă: Subprogram pentru trasarea graficului funcției  $f$ ,

$$f(x) = (x+1)(3-|x|), \quad x \in \mathbb{R}.$$

2°. Program BASIC (H.C.-85)

5 CLS

10 PRINT "REPREZENTAREA GRAFICĂ A FUNCȚIEI  $f, f(x) = (x+1)(3-|x|)$ "

20 PRINT "EXPLICITÎND MODULUL OBTÎNEM:

$$f(x) = \begin{cases} -x * x + 2 * x + 3, & x \geq 0 \\ x * x + 4 * x + 6, & x < 0 \end{cases}$$

40 PRINT AT 21,0; FLASH 1; "TASTAȚI ORICE" : PAUSE 0

50 CLS : BEEP 0 .1,5 : PLOT 30,30 : DRAW 200,0 : DRAW -3,3 : DRAW 0,-6 : DRAW 3,3

60 PLOT 130,9 : DRAW 0,165 : DRAW -3,-3 : DRAW 6,0 : DRAW -3,3 : PRINT AT 0,17; "y"; AT 16,29; "x"; AT 19,15; "0"

70 FOR I=-4 TO 3.2 STEP 1/120

79 IF I<0 THEN PLOT 130+15 \* I, 30+15 \* (I \* I + 4 \* I + 3) : GO TO 90

80 PLOT 130+20 \* I, 30+15 \* (I+1) \* ((3-ABS(I

90 NEXT I

93 PRINT AT 19,5; "(-3,0)"; AT 19,19; "(3,0)"; AT 12,11; "(0,3)"; AT 17,10; OVER 1; "(-1,0)"

95 FOR I=0 TO 10 : CIRCLE OVER 1; 130, 75, 3: NEXT I

100 STOP

1 REM

Program "FUNCȚIA"

## 18.3. Program „FUNCȚIA”

2 REM

1°. Tema: Subprogram pentru trasarea graficului funcției

$$f(x) = 1x * x - 5 * x + 6, \quad x \in \mathbb{R}$$

3 REM

2°. Program BASIC (HC-85)

10 CLS

20 PRINT "Reprezentarea grafică a funcției  $f$ ,  $f(x) = 1x * x - 5 * x + 6$ "

30 PRINT "Explicitînd modulul avem :

$$f(x) = \begin{cases} 1x * x - 5 * x + 6 \\ -x * x + 5 * x - 6 \end{cases}$$

40 PRINT AT 21,0; FLASH 1; "Tastați orice !" : PAUSE 0

50 CLS : PLOT 30,30 : DRAW 200,0 : DRAW -3,3 : DRAW 0,-6 : DRAW 3,3

60 PLOT 130,9 : DRAW 0,165 : DRAW -3,-3 : DRAW 6,0 : DRAW -3,3 : PRINT AT 0,17; "y"; AT 18,29; "x"; AT 19,15; "0"

70 FOR i=-0.5 TO 5.5 STEP 1/120

80 PLOT 110+20 \* i, 30+15 \* ABS (i \* i - 5 \* i + 6)

90 NEXT i

100 PRINT AT 19,18; "2"; AT 19,21; "3"

110 PRINT AT 21,0; "Funcția dată :  $f(x) = 1x * x - 5 * x + 6$ " : STOP



#### 18.4. Program „Funcția”

```

1 REM
2 Program 'FUNCTIA'

2 REM
1'. Tema: Subprogram pentru trasarea graficului functiei f,
f(x)=(|x|+1)*(3-|x|), xER
2 REM
2'. Program BASIC (HC-85)

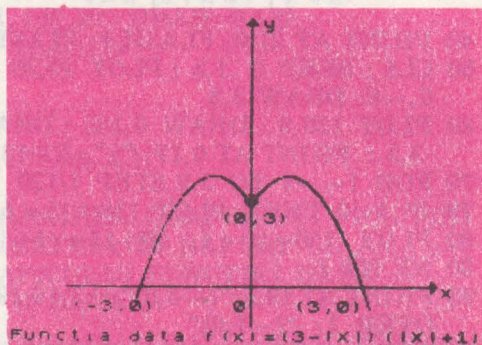
10 CLS
20 BEEP .2,1: PRINT : PRINT
Reprezentarea grafica a functiei- $f(x)=(|x|+1)*(3-|x|)$ 
3-10 BEEP .2,2: PRINT : PRINT
Explicativ modul de obtinere a urmatoarei forme a functiei:
ntru x>=0 f(x)=| -x*x+2*x+3,pe
ntru x<0 f(x)=| -x*x-2*x+3,pe
40 BEEP .2,3: PRINT : PRINT :
PRINT In continuare vom reprezenta cu ajutorul calculato-
rului functia data."
50 PRINT AT 21,0: FLASH 1;"Tas-
tati orice pentru a continua."
PAUSE 0
60 CLS : BEEP .1,5: BEEP .1,6:
BEEP .1,7: PLOT 30,30: DRAW 200,
0: DRAW -3,3: DRAW 0,-6: DRAW 3,
3
70 PLOT 130,9: DRAW 0,165: DRA
W -3,-3: DRAW 6,0: DRAW -3,3: PR
INT AT 0,17;"y";AT 18,29;"x";AT
19,15;"0"
80 FOR i=-3.2 TO 3.2 STEP 1/12

```

```

90 IF I<0 THEN PLOT 130+20*I,3
0+15*(-I*I-2*I+3)
100 IF I>=0 THEN PLOT 130+20*I,
30+15*(-I*I+2*I+3)
110 NEXT I
120 PRINT AT 19,4;"(-3,0)";AT 1
9,19;"(3,0)"; OVER 1;AT 13,14;"(
0,3)"
130 FOR I=0 TO 10: CIRCLE OVER
1,130,77,3: NEXT I
140 PRINT AT 21,0;"Funcția data
f(x)=3*(X-I)*(X+1)"
150 STOP

```



### 18.5. Program „Grafice – functii sinus, folosind modulul

**Tema:** Grafice de functii sinus folosind modulul:

$f(x)=\sin x$ ;  $f(x)=|\sin x|$ ;  $f(x)=\sin|x|$ ;  $f(x)=|\sin|x||$ ;  
modulația în amplitudine; amortizarea

```

10 PRINT AT 2,0;"Folosirea cal
culatorului pentru trasarea un
grafice ale unor functii SINUS
:
15 PRINT "1) 1)  $f(x) = \sin x$ 
2)  $f(x) = \sin x!$ 
3)  $f(x) = \sin|x|$ 
4)  $f(x) = \sin|x|!$ 
"
16 PRINT "5) Modulatia in amp
titudine: PRINT "  $y = A(1 + m \sin$ 
(w1*t))*SIN (w2*t)
17 PRINT "6) Relatia  $|y| = \sin$ 
x" 7) Functia de amorti
zare PERIODICA  $f(x) = e$ 
 $\uparrow (-x) * \sin x$ 
18 PRINT #1;" Apasati orice ta
sta!": PAUSE 0: CLS
20 PRINT "1) Graficul functiei
f:  $[-2\pi, 2\pi] \rightarrow [-1, 1]$ 
:
f(x) =  $\sin x$ 
30 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 126,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3

```

```

      35 PRINT AT 14,17;"0";AT 14,0;
      2*PI";AT 14,6;"PI";AT 12,23;"P
I";AT 12,29;"2*PI"
      37 FOR k=-40 TO 40 STEP 2: PLO
T 128+k,k+70: NEXT k
      39 PLOT 147,70: DRAW -9,9,PI/4
      : PRINT AT 12,18; OVER 1;"45°"
      40 FOR x=-2*PI TO 2*PI STEP .0
5
      50 LET y=SIN x
      60 PLOT x+18+128,y+18+70
      70 NEXT x
      80 PRINT #1;" Apasati orice ta
st "
      90 PAUSE 0: CLS
      120 PRINT "2) Graficul functiei
      : f(x)=sin x"
      : f(x)=sin x"
      130 PLOT 10,70: DRAW 240,0: DRA
W -5,-3: DRAW 0,6: DRAW 3,-3 PL
OT 128,10: DRAW 0,150: DRAW 3,-1

```



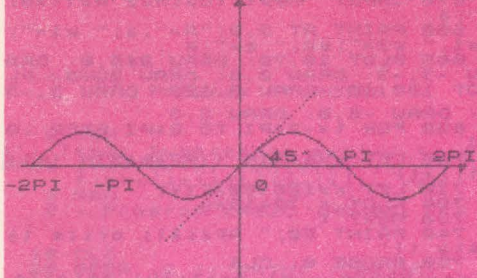
```

: DRAW -6,0: DRAW 3,3
135 PRINT AT 14,17;"0";AT 14,2;
"2PI";AT 14,6;"PI";AT 14,23;"P
I";AT 14,29;"2PI"
137 FOR k=-40 TO 40 STEP 2: PLO
T 128+k,k+70: NEXT k
139 PLOT 147,70: DRAW -9,9,PI/4
: PRINT AT 12,18: OVER 1;"45"
140 FOR x=-2*PI TO 2*PI STEP .0
5
150 LET y=ABS (SIN x)
160 PLOT x*18+128,y*18+70
170 NEXT x
180 PRINT #1;" Apasati orice ta
sta!"
190 PAUSE 0: CLS
220 PRINT "3) Graficul functiei
f: [-2PI,2PI] -> [-1,1]
f(x)=SIN |x|"
230 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
235 PRINT AT 14,17;"0";AT 14,0;
"-2PI";AT 14,6;"-PI";AT 14,23;"P
I";AT 14,29;"2PI"
237 FOR k=-40 TO 40 STEP 2: PLO
T 128+k,k+70: NEXT k
239 PLOT 147,70: DRAW -9,9,PI/4
: PRINT AT 12,18: OVER 1;"45"
240 FOR x=-2*PI TO 2*PI STEP .0
5
250 LET y=SIN (ABS x)
260 PLOT x*18+128,y*18+70
270 NEXT x
280 PRINT #1;" Apasati orice ta
sta!"
290 PAUSE 0: CLS
420 PRINT "4) Graficul functiei
f: [-2PI,2PI] -> [-1,1]
f(x)=|SIN |x||"
430 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
435 PRINT AT 14,17;"0";AT 14,0;
"-2PI";AT 14,6;"-PI";AT 14,23;"P
I";AT 14,29;"2PI"
437 FOR k=-40 TO 40 STEP 2: PLO
T 128+k,k+70: NEXT k
439 PLOT 147,70: DRAW -9,9,PI/4
: PRINT AT 12,18: OVER 1;"45"
440 FOR x=-2*PI TO 2*PI STEP .0
5
450 LET y=ABS (SIN x)
460 PLOT x*18+128,y*18+70
470 NEXT x
480 PRINT #1;" Apasati orice ta
sta!"
490 PAUSE 0: CLS
495 PRINT AT 0,0;"5) Modulatia
in amplitudine"
497 PRINT AT 2,0;"Cazuri concre
te:"
500 INPUT "Indicele de modulati
e: m=";m
510 INPUT "Pulsatia Semnalului
Modulator w1=";w1: INPUT "p
ulsatia PURTATOAREI w2=";w2
520 INPUT "Amplitudinea A=";a
525 CLS
530 PRINT AT 0,0;"5) Modulatia
in amplitudine"
540 PRINT AT 2,0;"A=";a;" w1="
;w1;" w2=";w2;" m=";m;" w1="
550 PLOT 10,70: DRAW 245,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,120: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
550 FOR t=-2*PI TO 2*PI STEP .0
1
570 LET y=a*(1+m*SIN (w1*t))*SI
N (w2*t)
575 IF ABS y>6.9 THEN GO TO 595
580 LET x=128+60/PI*t
590 PLOT x,70+10*y
600 NEXT t
610 PRINT #0;" Apasati orice ta
sta!"
620 PAUSE 0: CLS
625 GO TO 750
630 PRINT "6) Modulatia in FRE
CUENTA"
640 INPUT "Indicele de modulati
e m=";m
650 INPUT "Pulsatia purtatoarei
w1=";w1
660 INPUT "Pulsatia modulatora
e w2=";w2
670 INPUT "Amplitudinea A=(<=6)
";a
680 PRINT AT 2,2;"A=";a;" w1=";
w1;" w2=";w2;" m=";m
685 PLOT 10,70: DRAW 245,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,120: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
685 PRINT AT 6,15;"y";AT 12,30;
"x"
690 FOR x=-2*PI TO 2*PI STEP .0
5
700 IF SIN x<0 THEN NEXT x
810 LET y=SIN x
820 PLOT 18*x+128,70+18*y: PLOT
18*x+128,(-18*y)+70
830 NEXT x
840 PRINT #0;" Apasati orice ta
sta!"
850 PAUSE 0: CLS
855 PRINT "7) Functia de amorti
zare
perioadica f(x)=e^(-x)*SIN x"
860 PLOT 10,70: DRAW 205,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3
870 PLOT 40,10: DRAW 0,120: DRA
W 3,-3: DRAW -6,0: DRAW 3,3
880 FOR x=0 TO 4.6 STEP .01
890 LET y=EXP (-x)*SIN (3*x)
900 PLOT 40+30*x,140*y+70
930 NEXT x
940 FOR x=.4 TO 4.6 STEP .01
950 IF x>.8 THEN PLOT 40+30*x,-
140*EXP (-x)+70
960 PLOT 40+30*x,140*EXP (-x)+7
0
970 IF ABS (x-1)<.01 THEN PRINT
AT 7,10;"-- g(x)=e^(-x)"
980 IF ABS (x-1.5)<.01 THEN PR
INT AT 18,10;"-- h(x)=e^(-x)"
990 NEXT x
3) Graficul functiei:
f: [-2PI,2PI] -> [-1,1];
f(x)=SIN |x|
4) Graficul functiei:
f: [-2PI,2PI] -> [-1,1];
f(x)=|SIN |x||

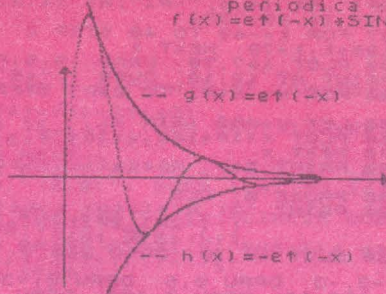
```



1) Graficul functiei:  
 $f: [-2\pi, 2\pi] \rightarrow [-1, 1]; f(x) = \sin x$

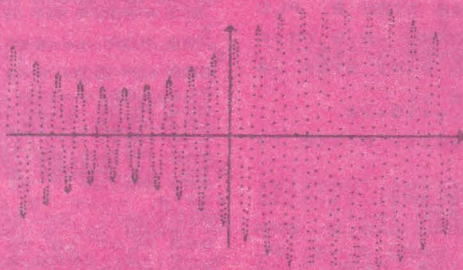


2) Functia de amortizare  
 periodica  
 $f(x) = e^{\pi(-x)} \sin x$



5) Modulatia in amplitudine

$A=5 \quad w_1=0.5 \quad w_2=10 \quad m=0.5$



## 18.6. Grafice – funcții sinus

Tema: Grafice ale unor funcții sinus.

10 PRINT AT 2,0;"Folosirea cal  
 culatorului pentru trasarea unor  
 grafice ale unor functii SINUS

15 PRINT "1) f(x) = sin x  
 2) f(x) = sin(2\*x)  
 3) f(x) = sin((1/2)\*x)  
 4) f(x) = sin(w\*t + c)

17 PRINT #1;" Apasati orice ta  
 sta!"; PAUSE 0: CLS  
 20 PRINT "1) Graficul functiei  
 f: [-2π, 2π] → [-1, 1]

f(x) = SIN x  
 30 PLOT 10,70: DRAW 240,0: DRA  
 W -3,-3: DRAW 0,6: DRAW 3,-3: PL  
 OT 128,10: DRAW 0,150: DRAW 3,-3  
 : DRAW -6,0: DRAW 3,3  
 35 PRINT AT 14,17;"0"; AT 14,0;  
 "-2π"; AT 14,7;"-π"; AT 14,22;"P  
 I"; AT 14,29;"2π"

38 FOR k=-40 TO 40 STEP 2: PLO  
 T 128+k,70+k: NEXT k  
 39 PLOT 142,70: DRAW -5,10,2:  
 PRINT AT 12,15, OVER 1;"45°"

40 FOR x=-2\*PI TO 2\*PI STEP .0  
 5  
 50 LET y=SIN x  
 60 PLOT x\*18+126,y\*18+70  
 70 NEXT x  
 80 PRINT #1;" Apasati orice ta  
 sta!"

90 PAUSE 0: CLS  
 120 PRINT "2) Graficul functiei  
 f: [-2π, 2π] → [-1, 1]

f(x) = SIN (2\*x)  
 130 PLOT 10,70: DRAW 240,0: DRA  
 W -3,-3: DRAW 0,6: DRAW 3,-3: PL  
 OT 128,10: DRAW 0,150: DRAW 3,-3  
 : DRAW -6,0: DRAW 3,3  
 135 PRINT AT 14,17;"0"; AT 14,0;  
 "-2π"; AT 14,7;"-π"; AT 14,22;"P  
 I"; AT 14,29;"2π"

140 FOR x=-2\*PI TO 2\*PI STEP .0  
 2  
 150 LET y=SIN (2\*x)

160 PLOT x\*18+126,y\*18+70  
 170 NEXT x

180 PRINT #1;" Apasati orice ta  
 sta!"  
 190 PAUSE 0: CLS  
 220 PRINT "3) Graficul functiei  
 f: [-2π, 2π] → [-1, 1]

f(x) = SIN (1/2)\*x  
 230 PLOT 10,70: DRAW 240,0: DRA  
 W -3,-3: DRAW 0,6: DRAW 3,-3: PL  
 OT 128,10: DRAW 0,150: DRAW 3,-3  
 : DRAW -6,0: DRAW 3,3

235 PRINT AT 14,17;"0"; AT 14,0;  
 "-2π"; AT 14,7;"-π"; AT 14,22;"P  
 I"; AT 14,29;"2π"

240 FOR x=-2\*PI TO 2\*PI STEP .0  
 5

250 LET y=SIN ((1/2)\*x)  
 260 PLOT x\*18+126,y\*18+70  
 270 NEXT x

280 PRINT #1;" Apasati orice ta  
 sta!"  
 290 PAUSE 0: CLS

420 PRINT "4) Graficul functiei  
 f: [-2π, 2π] → [-1, 1]

f(x) = SIN (w\*t + c)  
 425 INPUT "Introduceti pulsatia  
 w="; w: INPUT "Introduceti faza  
 initiala c="; c

427 PRINT AT 5,2;"w="; w; AT 6,2;  
 "c="; c

430 PLOT 10,70: DRAW 240,0: DRA  
 W -3,-3: DRAW 0,6: DRAW 3,-3: PL  
 OT 128,10: DRAW 0,150: DRAW 3,-3  
 : DRAW -6,0: DRAW 3,3

435 PRINT AT 14,17;"0"; AT 14,0;  
 "-2π"; AT 14,7;"-π"; AT 14,22;"P  
 I"; AT 14,29;"2π"

440 FOR x=-2\*PI TO 2\*PI STEP 1/  
 (w\*10)

450 LET y=SIN (w\*x+c)  
 460 PLOT x\*18+126,y\*18+70

470>NEXT x  
 480 PRINT #1;" Apasati orice ta  
 sta!"

490 PAUSE 0: CLS : PRINT AT 10,  
 13,



## 18.7. Program „Funcții cosinus”

Tema: Grafice de funcții cosinus, date prin:

$$f(x) = \cos x; f(x) = \cos 2x; f(x) = \cos \frac{1}{2}x; f(x) = \cos(\omega t + c)$$

```

10 PRINT AT 2,0;"Folosirea calculatorului pentru trasarea unor grafice ale unor funcții COSIN
US:"
15 PRINT "1) f(x) = cos x
2) f(x) = cos(2*x)
3) f(x) = cos((1/2)*x)
4) f(x) = cos(w*t+c)"
17 PRINT #1;" Apasati orice ta
sta!" PAUSE 0:CLS
20 PRINT "1) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=COS x
30 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
35 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 12,22;"P
I";AT 12,29;"2*PI"
38 FOR k=-40 TO 40 STEP 2: PLO
T 128-k+18*PI/2,+70+k: NEXT k
39 PLOT 142+18*PI/2,70: DRAW -
22,10,2: PRINT AT 10,18; OVER 1;
135
40 FOR x=-2*PI TO 2*PI STEP .0
5
50 LET y=COS x
60 PLOT x*18+128,y*18+70
70 NEXT x
80 PRINT #1;" Apasati orice ta
sta!"
90 PAUSE 0:CLS
120 PRINT "2) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=COS (2*x)
130 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
135 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 14,29;"2*PI"
140 FOR x=-2*PI TO 2*PI STEP .0
5
150 LET y=COS (2*x)
160 PLOT x*18+128,y*18+70
170 NEXT x
180 PRINT #1;" Apasati orice ta
sta!"
190 PAUSE 0:CLS
220 PRINT "3) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=COS ((1/2)*x)
230 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
235 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 14,29;"2*PI"
240 FOR x=-2*PI TO 2*PI STEP .05
250 LET y=COS ((1/2)*x)
260 PLOT x*18+128,y*18+70
270 NEXT x
280 PRINT #1;" Apasati orice ta
sta!"
290 PAUSE 0:CLS
420 PRINT "4) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=COS (w*t+c)
425 INPUT "Introduceti pulsatia w=";w: INPUT "Introduceti faza initiala c=";c
427 PRINT AT 5,2;"w=";w;AT 6,2;"c=";c
430 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
435 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 14,29;"2*PI"
440 FOR x=-2*PI TO 2*PI STEP 1/(w*10)
450 LET y=COS (w*x+c)
460 PLOT x*18+128,y*18+70
470 NEXT x
480 PRINT #1;" Apasati orice ta
sta!"
490 PAUSE 0:CLS: PRINT AT 10,
13;

```

## 18.8. Program „Funcții cosinus, folosind modulul”

Tema: Grafice de funcții cosinus, definite prin:

$$f(x) = \cos x; f(x) = |\cos x|; f(x) = |\cos x|; f(x) = |\cos x|$$

```

10 PRINT AT 2,0;"Folosirea calculatorului pentru trasarea unor grafice cosinusoidale:"
15 PRINT "1) f(x) = cos x
2) f(x) = |cos x|
3) f(x) = |cos x|
4) f(x) = |cos x|
"
17 PRINT #1;" Apasati orice ta
sta!" PAUSE 0:CLS
20 PRINT "1) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=COS x
30 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
35 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 12,22;"P
I";AT 12,29;"2*PI"
40 FOR x=-2*PI TO 2*PI STEP .0
5
50 LET y=COS x
60 PLOT x*18+128,y*18+70
70 NEXT x
80 PRINT #1;" Apasati orice ta
sta!"
90 PAUSE 0:CLS
120 PRINT "2) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=COS x
130 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
135 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 14,29;"2*PI"
140 FOR x=-2*PI TO 2*PI STEP .0
5
150 LET y=ABS (COS x)
160 PLOT x*18+128,y*18+70
170 NEXT x
180 PRINT #1;" Apasati orice ta
sta!"
190 PAUSE 0:CLS
220 PRINT "3) Graficul functiei f: [-2*PI,2*PI] -> [-1,1]"
f(x)=|COS x|
230 PLOT 10,70: DRAW 240,0: DR
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,150: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
235 PRINT AT 14,17;"0";AT 14,0;
"-2*PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 14,29;"2*PI"
240 FOR x=-2*PI TO 2*PI STEP .0
5
250 LET y=ABS (COS x)
260 PLOT x*18+128,y*18+70
270 NEXT x
280 PRINT #1;" Apasati orice ta
sta!"
290 PAUSE 0:CLS: PRINT AT 10,
13;

```



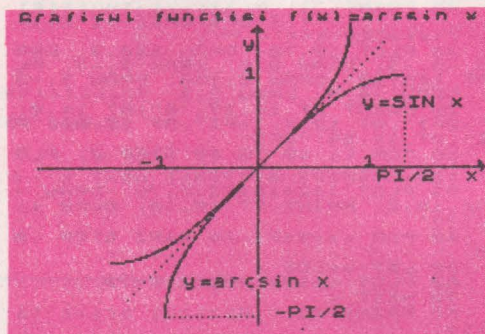
## 18.10. Program „arc cos”

Tema: Graficul funcției  $x \rightarrow \arcsin x$

```

5 PRINT "Graficul funcției f(x)=arcsin x"
6 PRINT AT 10,8; "-1"; AT 10,23
; "1"; AT 4,15; "1"
7 PLOT 10,88: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,5: DRAW 0,160: DRAW 3,-3:
DRAW -6,0: DRAW 3,3
8 REM FOR y=0 TO 25*PI STEP 3
: PLOT 78,88-y: PLOT 178,88+y: N
EXT y
9 FOR t=-70 TO 70 STEP 3: PLO
T 128+t,88+t: NEXT t
10 PRINT AT 2,15; "y"; AT 11,30;
"x"
15 FOR t=-PI/2 TO PI/2 STEP .0
1
20 LET y=SIN t
30 PLOT 128+50*t,88+50*y
35 PLOT 128+50*y,88+50*t
40 NEXT t
42 PRINT AT 11,24; "PI/2": FOR
i=0 TO 50 STEP 3: PLOT 128+25*PI
,88+i: NEXT i
43 PRINT AT 6,23; "y=SIN x"; AT
16,11; "y=arcsin x"
45 PRINT AT 20,17; "-PI/2": FOR
i=0 TO 50 STEP 3: PLOT 128-i,88
-25*PI: NEXT i

```



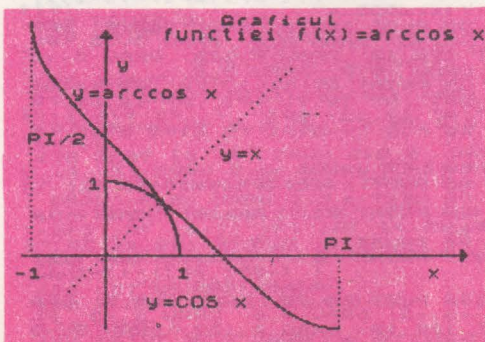
## 18.9. Program „arc sin”

Tema: Graficul funcției  $x \rightarrow \arccos x$

```

5 PRINT AT 0,14; "Graficul "; A
T 1,10; "funcției f(x)=arccos x"
7 PLOT 5,45: DRAW 240,0: DRAW
-3,-3: DRAW 0,6: DRAW 3,-3: PLO
T 50,5: DRAW 0,160: DRAW 3,-3: D
RAW -6,0: DRAW 3,3
10 PRINT AT 3,7; "y"; AT 17,26; "
x"
12 FOR t=-20 TO 100 STEP 3: PL
OT 50+t,45+t: NEXT t
16 FOR t=0 TO PI STEP .01
20 LET y=COS t
30 PLOT 50+40*t,45+40*y
35 PLOT 50+40*y,45+40*t
40 NEXT t
50 PRINT AT 17,11; "1"; AT 11,5;
"1"; AT 15,21; "PI"
55 PRINT AT 17,0; "-1": FOR i=0
TO 40*PI STEP 3: PLOT 10,45+i:
NEXT i: PRINT AT 6,1; "PI/2"
60 FOR i=0 TO 40 STEP 3: PLOT
50+40*PI,45-i: NEXT i
70 PRINT AT 9,14; "y=x"; AT 5,4;
"y=arccos x"; AT 19,9; "y=COS x"

```





## 18.11. Program „tangenta”

Tema: Graficul funcției  $x \rightarrow \operatorname{tg} x$ .

$$f(x) = \operatorname{tg} x$$

$$f(x) = |\operatorname{tg} x|$$

$$f(x) = \operatorname{tg}|x|$$

$$f(x) = |\operatorname{tg}|x||$$

```

10 PRINT AT 2,0;"Folosirea cal
culatorului pentru trasarea graf
icelor unor functii TANGENTA:"
15 PRINT "1) f(x)=tg x
2) f(x)=|tg x|
3) f(x)=tg|x|
4) f(x)=|tg|x||"
17 PRINT #1;" Apasati orice ta
sta!": PAUSE 0: CLS
20 PRINT "2) Graficul functiei
f: [-2PI,-3PI/2)U(-3P
I/2,-PI/2)U U(-PI/2,PI/2)U(PI/2
,3PI/2)U U(3PI/2,2PI)->R f
(x)=tg x"
30 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,130: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
135 PRINT AT 14,17;"0";AT 14,0;
"-2PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 12,29;"2PI"
36 FOR k=-60 TO 60 STEP 2: PLO
T 128-27*PI,k+80: NEXT k
37 FOR k=-60 TO 60 STEP 2: PLO
T 128-9*PI,k+80: NEXT k
38 FOR k=-60 TO 60 STEP 2: PLO
T 128+27*PI,k+80: NEXT k
39 FOR k=-60 TO 60 STEP 2: PLO
T 128+9*PI,k+80: NEXT k
40 FOR j=-23 TO 23 STEP 2: PLO
T 128+j,70+j: NEXT j
42 FOR x=-2*PI TO 2*PI STEP .0
2
50 LET y=TAN x
55 IF 18*ABS y>70 THEN GO TO 7
0
60 PLOT x*18+128,y*18+70
70 NEXT x
80 PRINT #1;" Apasati orice ta
sta!":
90 PAUSE 0: CLS
120 PRINT "3) Graficul functiei
f: [-2PI,-3PI/2)U(-3P
I/2,-PI/2)U U(-PI/2,PI/2)U(PI/2
,3PI/2)U U(3PI/2,2PI)->R f
(x)=|tg x|"
130 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,130: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
135 PRINT AT 14,17;"0";AT 14,0;
"-2PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 12,29;"2PI"
136 FOR k=-60 TO 60 STEP 2: PLO
T 128-27*PI,k+80: NEXT k
137 FOR k=-60 TO 60 STEP 2: PLO
T 128-9*PI,k+80: NEXT k
138 FOR k=-60 TO 60 STEP 2: PLO
T 128+27*PI,k+80: NEXT k
139 FOR k=-60 TO 60 STEP 2: PLO
T 128+9*PI,k+80: NEXT k
140 FOR j=-23 TO 23 STEP 2: PLO
T 128+j,70+j: NEXT j
142 FOR x=-2*PI TO 2*PI STEP .0
2
150 LET y=ABS (TAN x)

```

```

155 IF 18*ABS y>70 THEN GO TO 1
70
160 PLOT x*18+128,y*18+70
170 NEXT x
180 PRINT #1;" Apasati orice ta
sta!":
190 PAUSE 0: CLS
220 PRINT "3) Graficul functiei
f: [-2PI,-3PI/2)U(-3P
I/2,-PI/2)U U(-PI/2,PI/2)U(PI/2
,3PI/2)U U(3PI/2,2PI)->R f
(x)=tg |x|"
230 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,130: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
235 PRINT AT 14,17;"0";AT 12,0;
"-2PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 12,29;"2PI"
236 FOR k=-60 TO 60 STEP 2: PLO
T 128-27*PI,k+80: NEXT k
237 FOR k=-60 TO 60 STEP 2: PLO
T 128-9*PI,k+80: NEXT k
238 FOR k=-60 TO 60 STEP 2: PLO
T 128+27*PI,k+80: NEXT k
239 FOR k=-60 TO 60 STEP 2: PLO
T 128+9*PI,k+80: NEXT k
240 FOR j=-23 TO 23 STEP 2: PLO
T 128+j,70+j: NEXT j
242 FOR x=-2*PI TO 2*PI STEP .0
2
250 LET y=TAN (ABS x)
255 IF 18*ABS y>70 THEN GO TO 2
70
260 PLOT x*18+128,y*18+70
270 NEXT x
280 PRINT #1;" Apasati orice ta
sta!":
290 PAUSE 0: CLS
420 PRINT "4) Graficul functiei
f: [-2PI,-3PI/2)U(-3P
I/2,-PI/2)U U(-PI/2,PI/2)U(PI/2
,3PI/2)U U(3PI/2,2PI)->R f
(x)=|tg|x||"
430 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,130: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
435 PRINT AT 14,17;"0";AT 14,0;
"-2PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 12,29;"2PI"
436 FOR k=-60 TO 60 STEP 2: PLO
T 128-27*PI,k+80: NEXT k
437 FOR k=-60 TO 60 STEP 2: PLO
T 128-9*PI,k+80: NEXT k
438 FOR k=-60 TO 60 STEP 2: PLO
T 128+27*PI,k+80: NEXT k
439 FOR k=-60 TO 60 STEP 2: PLO
T 128+9*PI,k+80: NEXT k
440 FOR j=-23 TO 23 STEP 2: PLO
T 128+j,70+j: NEXT j
442 FOR x=-2*PI TO 2*PI STEP .0
2
450 LET y=ABS (TAN x)
455 IF 18*ABS y>70 THEN GO TO 1
70
460 PLOT x*18+128,y*18+70
470 NEXT x
480 PRINT #1;" Apasati orice ta
sta!":
490 PAUSE 0: CLS : PRINT AT 10,
13;"LOAD!": LOAD

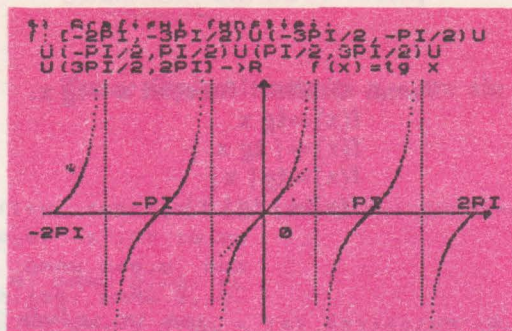
```



```

1) Graficul functiei:
f: (-2PI,-PI)U(-PI,0)U(0,PI)U(PI,2PI)U(2PI,3PI)U
U(-PI/2,PI/2)U(PI/2,3PI/2)U
U(3PI/2,3PI)U(3PI,4PI)U(4PI,5PI)U(5PI,6PI)U
f(x)=ctg x

```



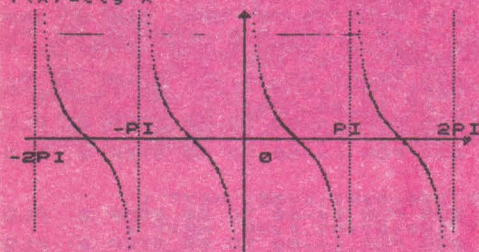
## 18.12. Program „cotangenta”

Tema: Graficul functiei  $x \rightarrow \text{ctg } x$

```

1) Graficul functiei:
f: (-2PI,-PI)U(-PI,0)U(0,PI)U(PI,2PI)U
U(2PI,3PI)U(3PI,4PI)U(4PI,5PI)U(5PI,6PI)U
f(x)=ctg x

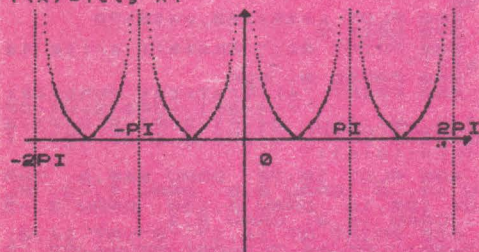
```



```

2) Graficul functiei:
f: (-2PI,-PI)U(-PI,0)U(0,PI)U(PI,2PI)U
U(2PI,3PI)U(3PI,4PI)U(4PI,5PI)U(5PI,6PI)U
f(x)=1/ctg x

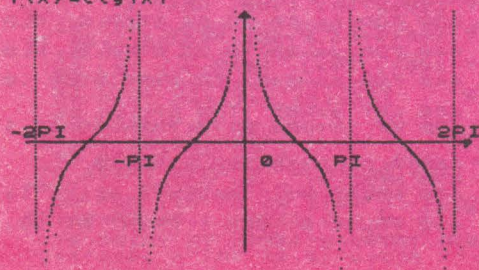
```



```

2) Graficul functiei:
f: (-2PI,-PI)U(-PI,0)U(0,PI)U(PI,2PI)U
U(2PI,3PI)U(3PI,4PI)U(4PI,5PI)U(5PI,6PI)U
f(x)=ctg |x|

```



```

150 LET y=1/ABS (TAN x)
155 IF 18*ABS y>70 THEN GO TO 1
160 PLOT x*18+126,y*18+70
170 NEXT x
180 PRINT #1;" Apasati orice ta

```

```

190 PAUSE 0: CLS
200 PRINT "3) Graficul functiei
f: (-2PI,-PI)U(-PI,0)
U(0,PI)U(PI,2PI)U(2PI,3PI)U(3PI,4PI)U(4PI,5PI)U(5PI,6PI)U
f(x)=ctg |x|
230 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,130: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
235 PRINT AT 14,17;"0";AT 12,0;
"-2PI";AT 14,7;"-PI";AT 14,22;"P
I";AT 12,29;"2PI"
236 FOR k=-60 TO 60 STEP 2: PLO
T 128-36*PI,k+80: NEXT k
237 FOR k=-60 TO 60 STEP 2: PLO
T 128-18*PI,k+80: NEXT k
238 FOR k=-60 TO 60 STEP 2: PLO
T 128+18*PI,k+80: NEXT k
239 FOR k=-60 TO 60 STEP 2: PLO
T 128+36*PI,k+80: NEXT k
242 FOR x=-2*PI+.01 TO 2*PI STE

```

```

P .02
250 LET y=1/TAN (ABS x)
255 IF 18*ABS y>70 THEN GO TO 2
260 PLOT x*18+126,y*18+70
270 NEXT x
280 PRINT #1;" Apasati orice ta
290 PAUSE 0: CLS
300 PRINT "4) Graficul functiei
f: (-2PI,-PI)U(-PI,0)
U(0,PI)U(PI,2PI)U(2PI,3PI)U(3PI,4PI)U(4PI,5PI)U(5PI,6PI)U
f(x)=1/ctg |x|
430 PLOT 10,70: DRAW 240,0: DRA
W -3,-3: DRAW 0,6: DRAW 3,-3: PL
OT 128,10: DRAW 0,130: DRAW 3,-3
: DRAW -6,0: DRAW 3,3
435 PRINT AT 14,17;"0";AT 14,0;
"-2PI";AT 12,7;"-PI";AT 12,22;"P
I";AT 12,29;"2PI"
436 FOR k=-60 TO 60 STEP 2: PLO
T 128-36*PI,k+80: NEXT k
437 FOR k=-60 TO 60 STEP 2: PLO
T 128-18*PI,k+80: NEXT k
438 FOR k=-60 TO 60 STEP 2: PLO
T 128+18*PI,k+80: NEXT k
439 FOR k=-60 TO 60 STEP 2: PLO
T 128+36*PI,k+80: NEXT k
442 FOR x=-2*PI TO 2*PI STEP .0
2
450 LET y=1/ABS (TAN x)
455 IF 18*ABS y>70 THEN GO TO 1
460 PLOT x*18+126,y*18+70
470 NEXT x
480 PRINT #1;" Apasati orice ta
490 PAUSE 0: CLS: PRINT AT 10,
13;"LOAD!": LOAD ""

```



## Metodica realizării programelor (exemplu)

Trebuie să postulăm ideea că în munca cu elevii, în cadrul laboratorului, unele subiecte care solicită alcătuirea de programe trebuie mai întâi discutate cu aceștia, stabilindu-se, pe cale orală, demersurile necesare realizării. În acest mod, elevii se vor obișnui să-și aprofundeze subiectul cercetat și să-și elaboreze prin studiu propriu programul pentru calculator.

Pentru exemplificare, vom presupune că se dorește realizarea unui program privind graficul funcției de gradul al doilea  $f: \mathbb{R} \rightarrow \mathbb{R}$ ,  $f(X) = AX^2 + BX + C$ ;  $A \neq 0$

Vom schița, împreună cu elevul Petrescu Iacob de la Liceul Industrial Dimitrie Cantemir din București (anul școlar 1983–1984) câteva cerințe ce stau în fața făuritorilor de programe, cerințe formulate într-o comunicare comună susținută, în 1984, la Brașov.

Evident, înainte de a concepe programul, elevul trebuie să cunoască funcția de gradul al doilea, conform prezentării acesteia în manualul de algebră pentru clasa a IX-a.

Apoi vom realiza o **organigramă**, corespunzătoare problemei date, care trebuie să evidențieze câteva etape strict necesare, pe care le vom enumera în continuare:

- 1) Introducerea coeficienților  $A$ ,  $B$ ,  $C$ .
- 2) Calculul rădăcinilor ecuației atașate, cu discutarea cazurilor particulare existente.
- 3) Afișarea valorilor rădăcinilor sau a altor mesaje corespunzătoare diferitelor cazuri.
- 4) Trecerea, opțională, la reprezentarea grafică.
- 5) Solicitarea domeniului în care urmează să se facă reprezentarea grafică.
- 6) Calculul coordonatelor vîrfului parabolei și afișarea lor;
- 7) Stabilirea ferestrei de afișare în cadrul ecranului dispozitivului de ieșire.
- 8) Calculul și reprezentarea funcției în cadrul domeniului dat.
- 9) Reluarea, opțională, a reprezentării grafice pentru alt domeniu de reprezentare.

În baza acestor cerințe se poate trece la scrierea programului. În final se pot fixa diferite triplete  $(A, B, C)$  de numere reale, drept coeficienți numerici ai funcției trinom, solicitându-se apoi graficele corespunzătoare acestora. Un exercițiu simplu și intuitiv îl constituie cazul tripletului  $(a, 0, 0)$ , unde  $a \in \{-1, 1\}$ .

Pentru aprofundarea acestui caz pot fi construite, în același sistem de axe de coordonate, familiile de parabole, prin introducerea numărului  $n$  al acestora ( $n \geq 2$ ). Spre exemplu, se pot lua mai întâi coeficienții:  $a=1$ ,  $a=1/2$ ,  $a=1/5$ ,  $a=1/10$  etc., scalarea reprezentării fiind cea pentru  $a=1$ .

De asemenea, se pot lua coeficienții:  $a=-1$ ,  $a=-1/2$ ,  $a=-1/5$  și  $a=-1/10$  etc., aprofundându-se astfel funcția de gradul al doilea prin variațiuni pe aceeași temă. Cazurile particulare pot fi extinse, desigur.

În conceperea și realizarea programului trebuie să ne preocupe și faptul că acesta trebuie să aibă un caracter conversațional, prin transmiterea către utilizator a numeroase mesaje ajutătoare, explicative și de eroare.

Listarea programului pe ecranul de vizualizare trebuie să permită celui ce-l va utiliza familiarizarea cu modulele conținute, evidențiindu-se structurile de intrare / ieșire, de calcul, precum și opțiunile oferite.

Pentru acest exemplu a fost aleasă funcția de gradul al doilea, deoarece trinomul  $AX^2 + BX + C$ ,  $A \neq 0$  poate fi înțeles în diverse ipostaze, generate de folosirea funcției modul, a funcțiilor "max" sau "min", a funcțiilor "partea întreagă" sau "partea fracționară", a funcțiilor "trigonometrice", "exponențiale" sau "logaritmice", a funcției "indicator de semn" etc.



## 18.13. Program „Inecuații și sisteme de inecuații”

— Cerc de matematică —

```
1 BORDER 1: PAPER 1: INK 7: C
LS
2 GO SUB 9900
10 FOR F=0 TO 7: PRINT PAPER 1
; INK F; AT 10,0;
```

MATEMATICE \*\*\*\* ORNAMENTE

```
11 IF INKEY$="" THEN NEXT F: G
O TO 10
15 CLS
6; FLASH 1; "ORNAMENTE MATEMATICE
```

25 PRINT "Ornament matematic

- un desen in care  
un anumit ansamblu de figur  
se repeta de un numar de or

30 PRINT "Desenului care sta  
la baza ornamentului ii es  
te caracteris-tica o ecuatie, o  
inecuatie, un sistem de ecuatii  
sau de inecuatii."  
35 GO SUB 9900: CLS  
40 PRINT FLASH 1; "EXEMPLUL 1";  
FLASH 0; "Reprezentind graf  
ic sistemul de inecuatii

$y - \sin x > 0$   
 $y + \sin x < 0$   
 $x \in [-2\pi, 2\pi]$

50 PRINT "constatam ca multim  
ea solutiilor inecuatiilor:

in x si y > s  
in x este formata din multime  
a punctelor domeniilor hasurate  
din figura urmatoare...

```
55 GO SUB 9900: CLS
60 GO SUB 9100
62 PRINT AT 0,0; "y-sinx>0  
y+sinx<0
```

xe[-2π, 2π]"  
65 PRINT AT 19,0; "sin x  
-sin x"

```
66 FOR f=0 TO 30 STEP 4: PLOT
50+f,10: PLOT 50+f,11: NEXT f
67 FOR f=0 TO 30: PLOT 50+f,20
: PLOT 50+f,21: NEXT f
70 FOR f=-2*PI TO 2*PI STEP .0
```

```
1 75 LET y=SIN f: PLOT 128+20*f,
20*y+88
76 NEXT f
80 FOR f=-2*PI TO 2*PI STEP .1
```

```
3 85 LET y=-SIN f: PLOT 128+20*f
,20*y+88
86 IF y>0 THEN DRAW 0,-40*y
90 NEXT f
```

95 GO SUB 9900: BEEP 1,20: CLS

100 PRINT FLASH 1; "EXEMPLUL 2";  
FLASH 0; "Reprezentind gra  
fic inecuatie:

(y - sinx)(y + sin  
x) < 0  
105 PRINT "pentru cazul y-sin  
x>0 y+sin  
x<0 avem regiunea hasura  
ta din

iar pentru cazul y-  
sin x<0  
sin x>0 avem regiunea nehasu  
rata din figura precedenta"  
110 FOR f=0 TO 7: PRINT INK f; A  
T 20,0; " Apasati -1- pentru a r  
eveni la imaginea precedenta"  
111 IF INKEY\$="1" THEN BEEP 1,-  
20: RUN 60  
112 IF INKEY\$="" THEN NEXT f: G

```
0 TO 110
115 BEEP 1,30: CLS
120 PRINT AT 10,10; "ASTFEL ..."  
: PAUSE 50: PRINT "Vom avea  
ornamentul din figura:" PAUSE 5  
0: CLS
```

130 PRINT AT 19,0; "sin x  
-sin x"

```
133 GO SUB 9100
134 PRINT AT 0,0; "(y-sin x)(y+  
sin x)<0"
```

```
135 FOR f=0 TO 30 STEP 4: PLOT
50+f,10: PLOT 50+f,11: NEXT f
140 FOR f=0 TO 30: PLOT 50+f,20
: PLOT 50+f,21: NEXT f
145 FOR f=-2*PI TO 2*PI STEP .0
```

```
9 150 LET y=SIN f: PLOT 128+20*f,
20*y+88
```

```
153 DRAW 0,-40*y
154 NEXT f
160 GO SUB 9900: CLS
```

190 BEEP 1,30  
200 PRINT FLASH 1; "EXEMPLUL 3";  
FLASH 0; "Multimea solutiilor  
inecuatiei:

```
210 PRINT "[(y-2/π*arcsin(sin(
π*x/2)))] * [(y+2/π*arcsin(sin(
π*x/2)))] * [(y-2/π*arcsin(sin(
π*(x-1)/2)))] * [(y+2/π*arcsin(sin(
π*(x-1)/2)))] <= 0"
```

215 PRINT "Este data de multim  
ea patratelor hasurate din figura  
urmatoare. Egalind fiecare fac  
tor cu 0 vom obtine ecuatiile li  
niilor frunte. Inegalitatea es  
te satisfacuta cind, fie unul d  
in cei 4 factori este negativ SI  
cei altii pozitivi, fie 3  
factori negativi si al patrulea  
pozitiv"

```
220 GO SUB 9900: CLS
223 PRINT AT 0,0; "[(y-2/π*arcsi  
n(sin(π*x/2)))] * [(y+2/π*arcsi
```

```
n(sin(π*x/2)))] * [(y-2/π*arcsi  
n(sin(π*(x-1)/2)))] * [(y+2/π*arcsi  
n(sin(π*(x-1)/2)))] <= 0"
```

```
225 PLOT 128,20: DRAW 0,110: DR  
AW -3,-7: DRAW 3,7: DRAW 3,-7: P  
LOT 0,89: DRAW 255,0: DRAW -7,4:
```

```
DRAW 7,-4: DRAW -7,-4  
230 PLOT 8,69
```

```
235 FOR f=0 TO 2  
240 DRAW 40,40: DRAW 40,-40  
245 NEXT f
```

```
250 PLOT 8,109  
255 FOR f=0 TO 2  
260 DRAW 40,-40: DRAW 40,40  
265 NEXT f
```

```
270 PLOT 8,89: DRAW 20,20  
275 FOR f=0 TO 1: DRAW 40,-40:  
DRAW 40,40: NEXT f: DRAW 40,-40:
```

```
DRAW 20,20  
280 PLOT 8,69: DRAW 20,-20  
285 FOR f=0 TO 1: DRAW 40,40: D  
RAW 40,-40: NEXT f: DRAW 40,40:
```

```
DRAW 20,-20  
290 PAUSE 100
```

```
295 FOR a=1 TO 240 STEP 40  
300 FOR f=0 TO 40 STEP 2  
301 PLOT f+8+a, f+69
```

```
305 IF f<10 THEN DRAW 0, (10-f)*2
```



```

310 IF f>10 AND f<20 THEN DRAW
0,-2*(f-10)
315 IF f>20 AND f<30 THEN DRAW
0,-2*(f-30)
320 IF f<40 AND f>30 THEN DRAW
0,(30-f)*2
325 IF f<40 AND f>30 THEN PLOT
f+8+a, f+49: DRAW 0, (30-f)*2
330 IF f>20 AND f<30 THEN PLOT
f+8+a, f+49: DRAW 0, -2*(f-30)
335 IF f>10 AND f<20 THEN PLOT
f+8+a, f+89: DRAW 0, -2*(f-10)
340 IF f<10 THEN PLOT f+8+a, f+8
9: DRAW 0, (10-f)*2
350 NEXT f
355 NEXT a
360 GO SUB 9000: BEEP 1,30
365 CLS
370 PRINT FLASH 1; AT 0,0; "EXEMP
LUL 4"; FLASH 0; "Multimea so
lutiilor inecuatiei:
x)]*
(x+pi/6)]*
(x-pi/6)]<0"
375 PRINT "este data de regi
unea hasurata din figura urmato
ra"
380 GO SUB 9000: CLS
385 PRINT AT 0,0; "y12-sint2(x)
J*
+pi/6)]*
-pi/6)]<0"
385 PLOT 126,20: DRAW 0,130: DR
AW -3,-7: DRAW 3,7: DRAW 3,-7: P
LOT 0,89: DRAW 255,0: DRAW -7,4:
DRAW 7,-4: DRAW -7,-4
400 FOR f=-2*PI TO 2*PI STEP .0
9: LET y=SIN f: LET a=126+20*f:
PLOT a,20*y+88: PLOT a,88-20*y
405 LET z=SIN (f-pi/6): PLOT a,
20*z+88: PLOT a,88-20*z
410 LET x=SIN (f+pi/6): PLOT a,
20*x+88: PLOT a,88-20*x
425 IF x<0 AND x>z THEN PLOT a,
20*x+88: DRAW 0,-40*x
430 IF z<0 AND z>x THEN PLOT a,
20*z+88: DRAW 0,-40*z
435 IF -z<0 AND -z>-x THEN PLOT
a,20*(-z)+88: DRAW 0,40*z
440 IF -x<0 AND -x>-z THEN PLOT
a,20*(-x)+88: DRAW 0,40*x
445 IF x<0 AND x>-y THEN PLOT
a,20*x+88: DRAW 0,-40*x
450 IF y<0 AND y>-z THEN PLOT
a,20*y+88: DRAW 0,-40*y: PLOT a,
20*y+98: DRAW 0,-40*y: PLOT a,20
*y+78: DRAW 0,-40*y
455 IF -y<0 AND -y>x THEN PLOT
a,20*-y+88: DRAW 0,40*y: PLOT a,
20*-y+98: DRAW 0,40*y: PLOT a,2
0*-y+78: DRAW 0,40*y
460 IF z>y AND z>-x AND -x<y TH
EN PLOT a,20*z+88: DRAW 0,20*(y-
z): PLOT a,20*-z+88: DRAW 0,20*(
z-y)
465 IF -z>-y AND -z>x AND x<-y
THEN PLOT a,20*-z+88: DRAW 0,20*
(z-y): PLOT a,20*z+88: DRAW 0,20
*(y-z)
470 IF x>y AND x>-z AND -z<y TH
EN PLOT a,20*x+88: DRAW 0,20*(y-
x): PLOT a,20*x+88: DRAW 0,20*(
x-y)

```

```

475 IF -x>-y AND -x>z AND z<-y
THEN PLOT a,20*-x+88: DRAW 0,20*
(x-y): PLOT a,20*-x+88: DRAW 0,
20*(y-x)
480 IF z<0 AND z>-y THEN PLOT
a,20*z+88: DRAW 0,-40*z
485 IF -z<0 AND -z>y THEN PLOT
a,20*-z+88: DRAW 0,40*z
490 IF -x<0 AND -x>y THEN PLOT
a,20*-x+88: DRAW 0,40*x
495 IF -y<0 AND -y>z THEN PL
OT a,20*-y+88: DRAW 0,40*y: PLOT
a,20*-y+98: DRAW 0,40*y: PLOT a,
20*-y+78: DRAW 0,40*y
500 IF y<0 AND y>-x THEN PLOT
a,20*y+88: DRAW 0,40*-y: PLOT a,
20*y+98: DRAW 0,40*-y: PLOT a,20
*y+78: DRAW 0,40*-y: PLOT a,20
550 NEXT f
560 GO SUB 9000: BEEP 1,30: BEE
P 1,20: BEEP 1,10: CLS
570 PRINT FLASH 1; AT 10,0; PAPE
R 3; INK 1; "PROGRAME REALIZATE
IN CADRUL
RMATICA "; INVERSE 1; "

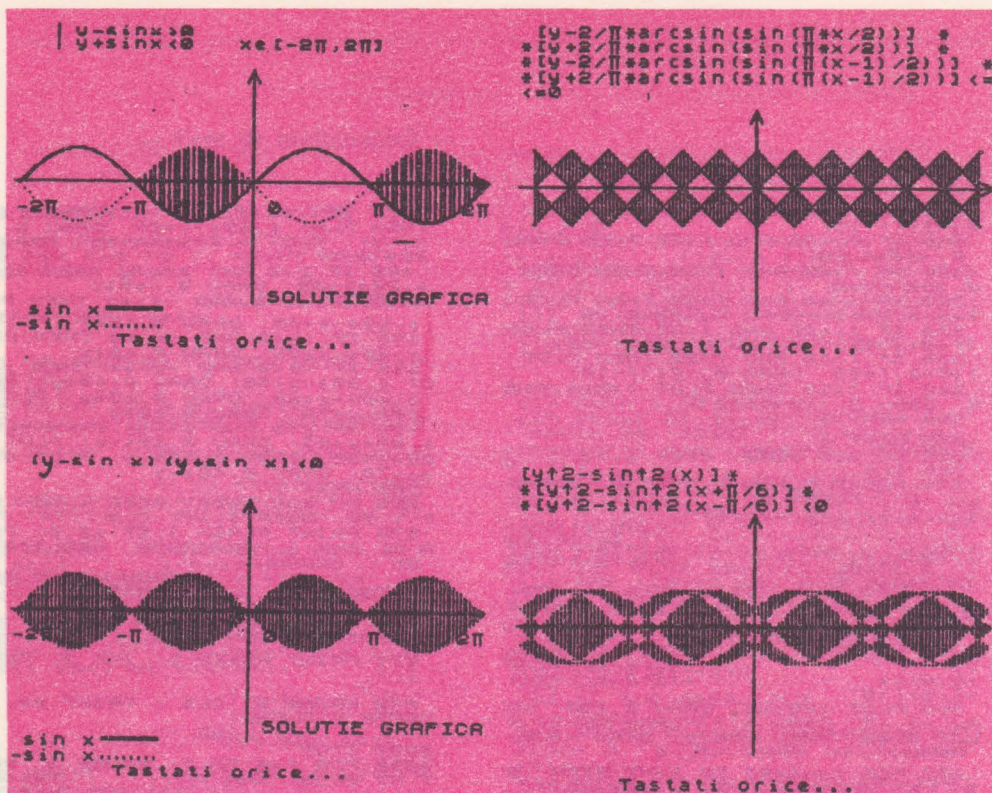
```

```

AL LICEULUI
DIMITRIE CANTEMIR
571 PRINT PAPER 6; INK 3; FLASH
1; "
572 PAUSE 0: CLS: PRINT #0; IN
VERSE 1; "LOAD
5999 STOP
9000 FOR f=0 TO 7: PAUSE 3: PRIN
T AT 21,0; INK f; "Tastati
forice...": IF INKEY$<>
"" THEN BEEP .2,-20: BEEP .2,-30
: RETURN
9001 NEXT f: GO TO 9000
9100 PLOT 126,20: DRAW 0,130: DR
AW -3,-7: DRAW 3,7: DRAW 3,-7: P
LOT 0,89: DRAW 255,0: DRAW -7,4:
DRAW 7,-4: DRAW -7,-4
9105 PRINT AT 12,0; OVER 1; INK
3; "21"
9107 PRINT AT 16,17: PAPER 5; IN
K 1; "SOLUTIE GRAFICA"
9110 RETURN
9980 LET L=USR "a"
9981 POKE L+0,BIN 1
9982 POKE L+1,BIN 1
9983 POKE L+2,BIN 1
9984 POKE L+4,BIN 1
9985 POKE L+5,BIN 1
9986 POKE L+6,BIN 1
9987 POKE L+7,BIN 1
9988 POKE L+3,BIN 1
9990 LET L=USR "p"
9991 POKE L+0,BIN 0
9992 POKE L+1,BIN 11111110
9993 POKE L+2,BIN 100100
9994 POKE L+3,BIN 100100
9995 POKE L+4,BIN 100100
9996 POKE L+5,BIN 100100
9997 POKE L+6,BIN 100100
9998 POKE L+7,BIN 100100
9999 RETURN

```





## 18.14. Program „Ecuatii de gradul II”

1°. Tema: Program de rezolvare a ecuațiilor de gradul al doilea în mulțimea  $\mathbb{C}$ .

2°. Program BASIC (HC-85) — „ECUAȚIA DE GRADUL II”

```
50 CLS
10 Print "PROGRAM DE REZOLVARE A ECUATIILOR DE GRADUL II"
20 INPUT " INTRODUCETI COEFICIENTII A,B,C "; A,B,C
30 IF A=0 THEN GO TO 100
40 LET D=B*B-4*A*C
50 IF D<0 THEN GO TO 80
60 LET X1=(-B+SQR(D))/(2*A) : LET X2=(-B-SQR(D))/(2*A)
70 PRINT " SOLUTIA : X1=" ; X1 ; " SI X2 = " ; X2 : STOP
80 LET R=(-B)/(2*A) : LET I=ABS ( SQR (-D)/(2*A) )
90 PRINT " AVEM : X1= " ; R ; " + i " ; I ; " SI X2 = " ; R ; " - i " ; I : STOP
100 PRINT " ECUATIA NU ESTE DE GRADUL II " : STOP
```



## 18.15. Program „Ecuția $a^x=N$ ”.

Tema: Rezolvarea grafică a ecuației  $a^x=N$ .

```

2 REM
  2' Program BASIC (HC-85)
5 CLS
10 PRINT
  1' Figurile (1) si (2) prezinta re-
  zolvarea ecuației  $a^x=N$  pentru
  cazurile:  $a>1$ ,  $0<a<1$ 
15 PRINT AT 21,0;"Graficele fu-
  ncției exponentiale."
20 PLOT 10,10: DRAW 115,0: DR
  AW 0,130: DRAW -115,0: DRAW 0,-13
  0
30 PRINT AT 5,2;"1";AT 5,29;"2"
40 PLOT 135,10: DRAW 115,0: DR
  AW 0,130: DRAW -115,0: DRAW 0,-1
  30
50 PLOT 60,10: DRAW 0,130
60 PLOT 185,10: DRAW 0,130
70 PLOT 10,35: DRAW 115,0
80 PLOT 135,35: DRAW 115,0
90 PRINT AT 5,6;"x";AT 5,22;"x"
  ;AT 16,14;"x";AT 16,30;"x";AT 1
  1,6;"N";AT 11,24;"N"
100 FOR X=0 TO 95 STEP .5
110 LET Y=1.05
120 LET Y=a^X
130 PLOT 10+X,35+Y
140 PLOT 245-X,35+Y
150 IF X=80 THEN PLOT 90,35: DR
  AW 0,49: DRAW -28,0: PLOT 166,35
  : DRAW 0,47: DRAW 20,0: PLOT 90,
  35: CIRCLE OVER 1;90,35,2: CIRCL
  E OVER 1;166,35,2

```

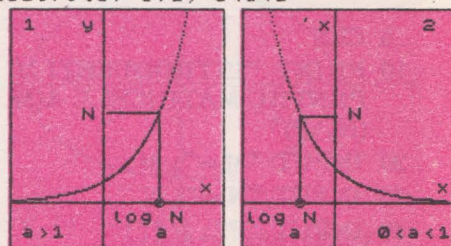
```

150 NEXT X
170 PRINT AT 19,2;"a>1";AT 19,2
  6;"0<a<1"
180 PRINT AT 18,8;"log N";AT 19
  ,11;"a";AT 16,17;"log N";AT 19,2
  0;"a"
190 STOP

```

Figurile (1) si (2) prezinta re-  
zolvarea ecuației  $a^x=N$  pentru

cazurile:  $a>1$ ,  $0<a<1$



Graficele funcției exponentiale.

## 18.16. Program „N factorial”

1>REM

1' Program "N FACTORIAL"

2 REM

1' Tema: CALCULUL FACTORIALULUI UNUI NUMAR NATURAL DAT

3 REM

2' Program BASIC (HC-85)

```

10 CLS
20 PRINT "PROGRAM DE CALCULARE
  A FACTORI-ALULUI UNUI NUMAR NA-
  TURAL N DAT"
30 PRINT
40 INPUT "INTRODUCETI N ";N
50 LET FACT=1
60 FOR K=1 TO N: PRINT AT 0,30
  ;K
70 LET FACT=FACT*K
80 NEXT K
90 PRINT "FACTORIALUL LUI ";N;
  " ESTE ";N;"!=";FACT
100 STOP

```

## 18.17. Program „Combinări”

1>REM

1' Program "COMBINARI"

2 REM

1' Tema: COMBINARI DE SA  
PTE NUMERE LUATE CITE TREI, NUME-  
RE APARTININD MULTIMII {1,2,3,4,  
5,6,7}

3 REM

2' Program BASIC (HC-85)

```

10 CLS
20>PRINT "SA SE TIPAREASCA TOA-
  TE COMBINA- RILE CARE SE POT FAC-
  E CU NUMERE-LE 1,2,3,4,5,6,7 LUC-
  TE CITE 3"
30 PRINT
40 LET A=1: LET B=2: LET C=3
50 PRINT "(";A;",";B;",";C;")"
60 IF C<7 THEN LET C=C+1: GO TO
  50
70 IF B<6 THEN LET B=B+1: GO TO
  50
80 IF A<5 THEN LET A=A+1: LET
  B=A: GO TO 50
90 STOP

```

## 18.19. Program „Transpusa”

Tema. Calculul transpusei unei matrici

A>REM

PROGRAM "TRANSPUSA"

1 REM

1' Tema: Calculul trans-  
pusei unei matrici

2 REM

2' Program BASIC (HC-85)

```

P
5 CLS
10 PRINT "TRANSPUSA UNEI MATRI-
  CI DE TIP M,N"
20 INPUT "INTRODUCETI M ";M
30 INPUT "INTRODUCETI N ";N
40 DIM A(M,N): DIM B(N,M)
50 FOR I=1 TO N
60 FOR J=1 TO M

```

```

70 INPUT "A(";I;");=";J;")=";
  A(I,J)
80 NEXT J
90 NEXT I
100 PRINT "Matricea data:"
110 FOR I=1 TO N
120 PRINT "FOR J=1 TO M
130 PRINT A(I,J);" ";
140 NEXT J
150 PRINT
160 NEXT I
170 FOR I=1 TO N
180 FOR J=1 TO M
190 LET B(J,I)=A(I,J)
200 NEXT J
210 NEXT I
220 PRINT "Matricea transpusa:"
230 FOR I=1 TO M
240 PRINT "FOR J=1 TO N
241 PRINT B(I,J);" "("
242 NEXT J: PRINT "NEXT I
243 STOP

```

18.18 la pag. 152



## 18.18. Program „PRODUS MAT”

1°. Tema: Produsul a două matrice

2°. Schema bloc:

3°. Program BASIC (H.C-85)

```

5 CLS
10 PRINT "INTRODUCETI M, N, P
    PENTRU MATRICELE A(M, N) si B
    (N, P)"
20 INPUT "M="; M
30 INPUT "N="; N
40 INPUT "P="; P
50 DIM A (M, N) : DIM B (N, P): DIM C (M, P)
60 FOR I=1 TO M
70 FOR J=1 TO N
80 PRINT "A("; I; J;")=";: INPUT
    A(I, J): PRINT A(I, J)
90 NEXT J
100 NEXT I
110 FOR I=1 TO N
120 FOR J=1 TO P
130 PRINT "B("; I; J; ")=";: INPUT
    B(I, J): PRINT B(I, J)
140 NEXT J
150 NEXT I
160 FOR I=1 TO M
170 FOR J=1 TO P
175 LET C(I, J)=0
180 FOR K=1 TO N
190 LET C(I, J)=C(I, J)+A(I, K) * B(K, J)
200 NEXT K
210 NEXT J
220 NEXT I
230 FOR I=1 TO M
240 FOR J=1 TO P
250 PRINT C(I, J); " ";
260 NEXT J
270 PRINT : PRINT
280 NEXT I
290 STOP
    
```

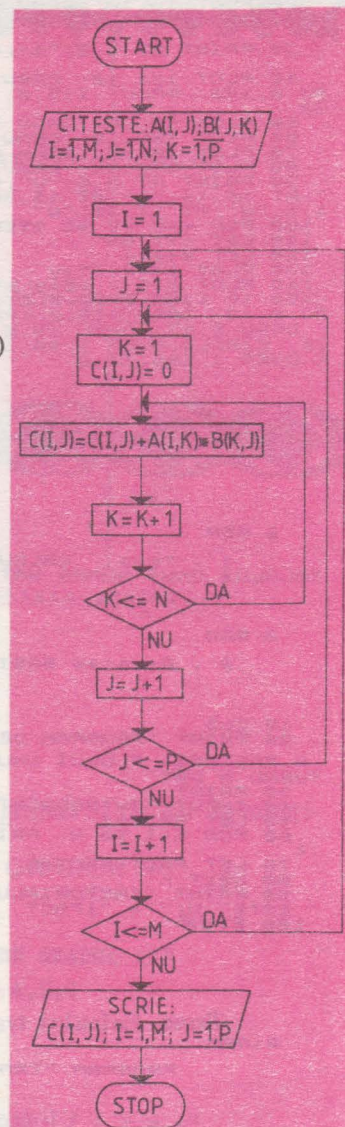


Fig. 18.1. Schema logică.  
Produs matrice



## 18.20. Program „Trasarea elipsei”

**Tema: Trasarea unei elipse**

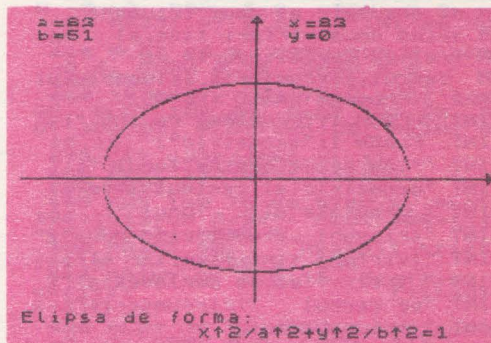
1 REM

2. Program 'TRASARE ELIPSA

1. Tema: Trasarea unei elipse. 1  
2. Programul in BASIC (HC-85)

```
10 REM ELIPSA
20 PRINT AT 20,0;"Elipsa de fo
rma:
x^2/a^2+y^2/b^2=1"
30 PLOT 127,20: DRAW 0,155: DR
AW -3,-3: DRAW 6,0: DRAW -3,3: P
LOT 0,87: DRAW 255,0: DRAW -3,3:
DRAW 0,-6: DRAW 3,3
40 INPUT "a=";a
50 IF a<0 OR a>125 THEN GO TO
40
60 INPUT "b=";b
70 IF b<0 OR b>65 THEN GO TO 4
0
80 INPUT "Pasul de desenare=";
s
90 PRINT AT 0,1;"a=";a;AT 1,1;
"b=";b
100 FOR k=-a TO a STEP s
110 LET y=b/a*SQR (a*a-k*k)
120 PRINT AT 0,18;"x="
";AT 1,18;"y="
130 PRINT AT 0,18;"x=";k;AT 1,1
8;"y=";y
```

```
140 IF ABS y>80 THEN GO TO 160
150 PLOT k+127,y+87: PLOT k+127
,y+87
160 NEXT k
```



## 18.21. Program „Trasare hiperbolă”

**Tema: Trasarea unei hiperbole.**

1 REM

OLA' Program 'TRASARE HIPERB

1. Tema: Trasarea unei hiperbole  
2. Programul in BASIC (HC-85)

```
10 REM HIPERBOLA
20 PRINT AT 20,0;"Hiperbola de
forma:
x^2/a^2-y^2/b^2=1"
30 PLOT 127,20: DRAW 0,155: DR
AW -3,-3: DRAW 6,0: DRAW -3,3: P
LOT 0,87: DRAW 255,0: DRAW -3,3:
DRAW 0,-6: DRAW 3,3
40 INPUT "a=";a
50 IF a<0 OR a>125 THEN GO TO
40
60 INPUT "b=";b
70 IF b<0 OR b>65 THEN GO TO 4
```

```
0 80 INPUT "Pasul de desenare=";
s
90 PRINT AT 0,1;"a=";a;AT 1,1;
"b=";b
100 LET t=SQR ((a*a*(65*65+b*b)
)/(b*b))
110 FOR k=-t TO t STEP s
120 IF k>=a AND k<=a THEN GO T
O 180
130 LET y=b/a*SQR (-a*a+k*k)
140 PRINT AT 0,18;"x="
";AT 1,18;"y="
150 PRINT AT 0,18;"x=";k;AT 1,1
8;"y=";y
160 IF ABS k>120 OR ABS y>65 TH
EN GO TO 180
170 PLOT k+127,y+87: PLOT k+127
,y+87
180 NEXT k
```

## 18.22. Program „Trasare parabolă”

**Tema: Trasarea unei parabole**

1 REM

2. Program 'TRASARE PARABO

1. Tema: Trasarea unei parabole. 1  
2. Programul in BASIC (HC-85)

```
10 REM PARABOLA
20 PRINT AT 20,0;"Parabola de
forma:
y^2=2*p*x"
30 PLOT 127,20: DRAW 0,155: DR
AW -3,-3: DRAW 6,0: DRAW -3,3: P
LOT 0,87: DRAW 255,0: DRAW -3,3:
DRAW 0,-6: DRAW 3,3
40 INPUT "p=";p
50 IF p<0 OR p>125 THEN GO TO
40
60 INPUT "Pasul de desenare=";
s
```

```
70 PRINT AT 0,1;"p=";p
80 FOR k=0 TO 120 STEP s
90 LET y=SQR (2*p*k)
100 PRINT AT 0,18;"x="
";AT 1,18;"y="
110 PRINT AT 0,18;"x=";k;AT 1,1
8;"y=";y
120 IF ABS y>65 THEN GO TO 140
130 PLOT k+127,y+87: PLOT k+127
,y+87
140 NEXT k
```



## 18.23. Program „Trasare loc geometric”

**Tema:** Trasarea locului geometric al centrului de greutate al unui triunghi având două vîrfuri fixe, al treilea deplasîndu-se pe cercul circumscris triunghiului.

```

1 REM 22) Program "TRASARE LOC "
2 REM "GEOMETRIC"
3 REM 1'.. Tema: Determinarea
4 REM locului geometric al centrului
5 REM de greutate al unui triunghi
6 REM avînd doua vîrfuri fixe, al
7 REM treilea deplasîndu-se pe
8 REM cercul circumscris triunghiului
9 REM 2'.. Program BASIC (HC-85)
10 PRINT FLASH 1;"Problema de loc ";
11 PRINT FLASH 1;"geometric"
12 PRINT "Isa se determine locul ";
13 PRINT "geometric al centrului de ";
14 PRINT "greutate al lunui ";
15 PRINT "triunghi ,atunci cînd doua";
16 PRINT "vîrfuri sint fixe,iar al";
17 PRINT "trei- ilea se deplaseaza ";
18 PRINT "pe cercul circumscris";
19 PRINT "triunghiului."
20
21 PRINT "Programul este structurat";
22 PRINT " pe doua idei: ";
23 PRINT " 1) Rezolvarea";
24 PRINT " analitica 2)";
25 PRINT " Demonstrarea grafica a ";
26 PRINT " calculatorului";
27 PRINT AT 21,0;"Tastati";
28 PRINT " orice pentru a continua!";
29 PAUSE 0
30
31 CLS : PRINT "1) Demonstrarea "
32 PRINT "analitica"
33 PLIN 200,150: DRAW -30,-60
34 DRAW 80,10: DRAW -50,50
35 PRINT AT 2,25;"A";AT 10,20;"B"
36 PRINT AT 2,25;"A";AT 10,20;"B"
37 PRINT AT 9,31;"C": DRAW 10,-55
38 PRINT AT 9,25;"M": PLOT 170,90
39 DRAW 60,35: PRINT AT 6,29;"N"
40 PRINT AT 8,26; OVER 1;"G"
41 PAUSE 100
42 PRINT AT 2,0;"Coordonatele"
43 PRINT " vîrfurilor sint:"
44 PRINT "A(x1,y1) ,B(x2,y2)"
45 PRINT " si C(x3,y3)"
46 PAUSE 100: PRINT "Avem relatia:";
47 PRINT " AG/GM=2"
48 PRINT "Coordonatele punctului"
49 PRINT "M sint:Xm=(X2+X3)/2 "
50 PRINT "si Ym=(Y2+Y3)/2"
51 PRINT "Deci Xg=(X1+X2+X3)/3 "
52 PRINT "si Yg=(Y1+Y2+Y3)/3-iese";
53 PRINT " imediat"
54 PRINT AT 11,18; OVER 1;"="
55 PRINT "Fie O(Xo,Yo) centrul cercului";
56 PRINT " circumscris si vom avea:";
57 PRINT " X1=Xo+Rcos t ,Y1=Yo";
58 PRINT "+Rsin t "
59 PRINT "Atunci: Xg=(Xo+X2+X3)/3+";
60 PRINT "R/3COS t Yg=(Yo+Y2+Y3)/3+";
61 PRINT "R/3SIN t."
62 PAUSE 100
63
64 PRINT "INVERSE 1;"Locul geometric";
65 PRINT " este ,deci, cercul de";
66 PRINT " raza R/3 si centru , ";
67 PRINT "centrul de greutate al ";
68 PRINT "triunghiului OBC"
69 CIRCLE 205,101,10
70 PRINT #1; FLASH 1;"Tastati orice";
71 PAUSE 0
72 CLS
73 PRINT AT 0,0;"2) Verificarea"
74 PRINT " grafica "
75 DIM x(63): DIM y(63)
76 CIRCLE 178,88,60
77 PLOT 130,52
78 PRINT AT 15,15;"B"
79 DRAW 98,3
80 PRINT AT 15,29;"C"
81 FOR t=0 TO 2*PI STEP 0.1
82 IF t>=5.8 THEN PRINT AT 15,29;"C"
83 LET x=178+60*COS t
84 LET y=88+60*SIN t
85 LET k=10*t+1
86 LET x(k)=178+20*COS t
87 LET y(k)=88+20*SIN t
88 PLOT x,y: DRAW 130-x,52-y
89 DRAW 98,3: DRAW x-228,y-55
90 PRINT AT 10-9*SIN t,21+9*COS t;"A"
91 PLOT x,y: DRAW 179-x,53-y
92 PLOT 130,52
93 DRAW (x-50)/2+9,y/2-25
94 FOR i=1 TO k: PLOT x(i),y(i)
95 NEXT i
96 CIRCLE 178,88,60
97 PLOT x,y
98 DRAW INVERSE 1;130-x,52-y
99 PLOT 228,55
100 DRAW INVERSE 1;x-228,y-55
101 PRINT AT 10-9*SIN t,21+9*COS t;" "
102 PLOT x,y
103 DRAW INVERSE 1;179-x,53-y
104 PLOT 130,52
105 DRAW INVERSE 1;(x-50)/2+9,y/2-25
106 NEXT t
107 PRINT AT 11,30;"A": PLOT 238,80
108 DRAW -108,-26: PLOT 238,80
109 DRAW -10,-23
110 FOR i=1 TO 20: PRINT AT i,12;"*"
111 NEXT i
112 PRINT AT 2,0;"Raspuns:"
113 PRINT AT 4,0;"Locul Geo-";
114 PRINT AT 5,0;"metric este"
115 PRINT "cercul din: PRINT "figura."
116 PAUSE 200: PRINT "Centrul sau"
117 PRINT "este cen-"
118 PRINT "de greutate al"
119 PRINT AT 12,0;"triunghiul-"
120 PRINT AT 13,0;"lui OBC"
121 PAUSE 100: PLOT 178,88: DRAW 1,0
122 DRAW 0,2: DRAW -2,0: DRAW 0,-2
123 DRAW 1,0
124 PRINT AT 10,21; FLASH 1;"O"
125 PRINT AT 15,15; FLASH 1;"B"
126 PRINT AT 15,29; FLASH 1;"C"
127
128 PLOT 178,65: DRAW 1,0: DRAW 0,1: DRAW -1,0: DRAW 0,-1: PAUSE 100
129 BEEP 1,6: PAUSE 50
130 PRINT AT 10,21;"O"
131 PRINT AT 10,21;"O"

```



## 18.24. Program „Grafice – funcții exponențiale și logaritmice”

```

10 PRINT AT 2,2;"CUPRINS:"; PR
INT AT 4,1;"1) Folosirea calcul
atorului pentru trasarea grafi
celor unor functii. EXPONENTIALA
..20 PRINT AT 8,1;"2) Folosirea
calculatorului pentru trasar
ea graficelor unor functii LOGAR
ITMICE.....2"
22 PRINT AT 12,2;"Studiu:
Variatia si Gr
aficul, Continuitatea,
Derivabilitate
a."
25 PRINT #0;"Apasati tasta cor
espunzatoare"
30 LET k$=INKEY$: IF k$="" THE
N GO TO 30
40 IF k$="2" THEN GO TO 840
50 CLS: PRINT AT 3,1;"1.Grafi
cul functiei: f(x)=3^x"
60 PLOT 40,5: DRAW 215,0: DRAW
-3,3: DRAW 0,-6: DRAW 3,3
70 PLOT 160,1: DRAW 0,140: DRA
W -3,-3: DRAW 6,0: DRAW -3,3
80 PRINT AT 5,19;"y";AT 20,30;
"x"
90 FOR x=-.5 TO 3.2 STEP .03
100 LET y=3^x
110 PLOT 128+25.6*x,4*y+5
120 IF x<-2.5 THEN PLOT 128+25.
6*x,6
130 IF ABS (x-1.1)<0.03 THEN PR
INT AT 18,19;"1"
140 IF x<0 THEN PRINT AT 10,0;"
Graficul functiei": PRINT AT 11,
0;"creste LENT!!!"
150 IF x>1.5 THEN PRINT AT 11,0
"creste BRUSC!!!"
160 BEEP .01,y
170 NEXT x
180 PRINT #0;" Apasati oric
e tasta"
190 PAUSE 0
200 CLS
210 PRINT AT 2,0;"2.Graficul fu
nctiei: f(x)=3^|x|"
220 PLOT 40,5: DRAW 215,0: DRAW
-3,3: DRAW 0,-6: DRAW 3,3
230 PLOT 160,1: DRAW 0,140: DRA
W -3,-3: DRAW 6,0: DRAW -3,3
240 PRINT AT 5,19;"y";AT 20,30;
"x"
250 FOR x=-3 TO 3 STEP .03
260 LET y=3^ABS x
270 IF ABS x<.018 THEN PRINT AT
0,3;" PUNCT DE MINIM A(0,1)!"
BEEP .5,40: PAUSE 100: PRINT AT
0,0;" "
280 PLOT 160+25.6*x,4*y+5
290 IF ABS (x-.2)<0.03 THEN PRI
NT AT 19,19;"1"
300 IF x<-5 THEN PRINT AT 5,0;
"Graficul functiei": PRINT AT 6,
0;"descresce BRUSC!!!"
310 IF x>-.5 AND x<0 THEN PRINT
AT 5,0;"Graficul functiei": PRI
NT AT 6,0;"descresce LENT!!!"
320 IF x<.5 AND x>0 THEN PRINT
AT 5,0;"Graficul functiei": PRIN
T AT 6,0;"creste LENT!!!"
330 IF x>.5 THEN PRINT AT 6,0;"
creste BRUSC!!!"
340 BEEP .01,y
350 NEXT x
360 FOR I=1 TO 70 STEP 3: PLOT
160+I,7*I/25.6+9: NEXT I
370 FOR I=1 TO -70 STEP -3: PLO
T 160+I,-7*I/25.6+9: NEXT I
380 PRINT #0;" Apasati oric
e tasta"
390 PAUSE 0
400 CLS
405 GO TO 690
410 CLS: PRINT AT 2,0;"4.Grafi
cul functiei:
f(x)=3^(|x|-1)"

```

```

420 PLOT 128,1: DRAW 0,140: DRA
W -3,-3: DRAW 6,0: DRAW -3,3
430 PRINT AT 5,14;"y";AT 20,30;
"x"
440 PLOT 20,5: DRAW 235,0: DRAW
-3,3: DRAW 0,-6: DRAW 3,3
450 FOR x=-2 TO 2 STEP .01
460 IF ABS x<.01 THEN PLOT 127,
25: PLOT 128,25: PLOT 129,25: PL
OT 130,25: PLOT 126,25: PRINT AT
18,15;"1"
470 LET y=3^((ABS x)-1)
480 PLOT 128+60*x,5+20*y
490 NEXT x
495 FOR t=0 TO 80 STEP 2: PLOT
128+t,11+LN 3/9*t: PLOT 128-t,11
+LN 3/9*t: NEXT t
500 PRINT #0;" Apasati orice
tasta !"
510 PRINT AT 2,0;"5. Graficul f
unctiei: f(x)=3^(|
x|-1)"
520 PLOT 70,1: DRAW 0,140: DRAW
-3,-3: DRAW 6,0: DRAW -3,3
530 PRINT AT 5,9;"y";AT 20,30;"
x"
540 PLOT 20,5: DRAW 235,0: DRAW
-3,3: DRAW 0,-6: DRAW 3,3
550 FOR x=-1 TO 2.8 STEP 0.01
555 IF ABS (x-1)<.01 THEN PRINT
AT 21,16;"1"
560 LET y=3^ABS (x-1)
570 PLOT 70+60*x,5+10*y
580 NEXT x
585 FOR t=0 TO 100 STEP 3: PLOT
130+t,14+1/4*t: PLOT 130-t,14+1
/4*t: NEXT t
590 PRINT #0;" Apasati orice t
asta !"
600 PRINT AT 2,0;"6. Graficul f
unctiei: f(x)=3^(|
|x|-1)|"
610 PLOT 128,1: DRAW 0,140: DRA
W -3,-3: DRAW 6,0: DRAW -3,3
620 PRINT AT 5,15;"y";AT 20,30;
"x"
630 PLOT 20,5: DRAW 235,0: DRAW
-3,3: DRAW 0,-6: DRAW 3,3
640 FOR x=-3 TO 3 STEP .01
650 LET y=3^ABS (ABS x-1)
660 PLOT 128+40*x,5+10*y
667 IF ABS (x-1)<.005 THEN PRIN
T #0;" -1 1"
670 NEXT x
680 PRINT #0;" Apasati orice t
asta !"
685 GO TO 840
690 PRINT AT 2,0;"3.Graficul fu
nctiei: f(x)=3^|-x|"
700 PLOT 40,5: DRAW 215,0: DRAW
-3,3: DRAW 0,-6: DRAW 3,3
710 PLOT 160,1: DRAW 0,140: DRA
W -3,-3: DRAW 6,0: DRAW -3,3
720 PRINT AT 5,19;"y";AT 20,30;
"x"
730 FOR x=-2.7 TO 2.7 STEP .03
740 LET y=3^(-ABS x)
750 IF x<0.018 AND x>-.018 THEN
PRINT AT 0,3;" PUNCT DE MAXIM A
(0,1)!"
BEEP .5,40: PAUSE 100:
PRINT AT 0,0;" "
760 PLOT 160+25.6*x,30*y+5
770 IF ABS (x-1.1)<0.03 THEN PR
INT AT 19,19;"1"
780 IF x<0 THEN PRINT AT 5,0;"G
raficul functiei": PRINT AT 6,0;
"creste LENT!!!"
790 IF x>0 THEN PRINT AT 6,0;"d
escresce LENT!!!"
800 BEEP .01,y
810 NEXT x
820 PRINT #1;" Apasati orice!"
830 PAUSE 0
835 GO TO 410
840 CLS: PRINT "FOLOSIREA CALC
ULATORULUI PENTRU TRASAREA GRAFI
CELOR UNOR FUNCTII LOG

```



```

ARITHMICE "
850 PRINT AT 3,0;"1.Graficul fu
nctiei f(x)=log(x)";AT 4,28;"2"
860 PLOT 20,100: DRAW 225,0: DR
AW -3,3: DRAW 0,-6: DRAW 3,3
870 PLOT 30,1: DRAW 0,140: DRAW
-3,-3: DRAW 6,0: DRAW -3,3
880 PRINT AT 5,4;"y";AT 8,29;"x"

890 FOR x=.02 TO 2.8 STEP .01
900 LET y=LN x/LN 2
910 PLOT 30+75*x,18*y+100
920 IF ABS (x-1)<.01 THEN PRINT
AT 5,12;"1"
930 IF x<1 THEN PRINT AT 13,13;
"Graficul functiei";AT 14,13;"Cr
este BRUSC!!!"
940 IF x>1 THEN PRINT AT 13,13;
"Graficul functiei";AT 14,13;"Cr
este LENT!!!"
950 BEEP .01,5*y+30
960 NEXT x

970 PRINT #0;" Apasati o tasta
!";PAUSE 0:CLS
980 PRINT AT 2,0;"2.Graficul fu
nctiei f(x)=log x";AT 3,29;"2"
990 PLOT 10,10: DRAW 240,0: DRA
W -3,3: DRAW 0,-6: DRAW 3,3: PLO
T 11,9: DRAW 0,140: DRAW 3,-3: D
RAW -6,0: DRAW 3,3
1000 PRINT AT 4,0;"y": PRINT AT
21,30;"x"
1010 FOR x=0.1 TO 3.3 STEP .01
1020 LET y=ABS (LN x)
1030 PLOT 10+70*x,10*y+60
1040 IF x<1 THEN PRINT AT 6,15;"
Graficul functiei";AT 7,15;"Desc
reste BRUSC!!!"
1050 IF x>1 THEN PRINT AT 6,15;"
Graficul functiei";AT 7,15;"Cres
te LENT!!!"
1060 IF ABS (x-1)<.01 THEN PRINT
AT 21,9;"1"
1070 BEEP .01,1+10*y

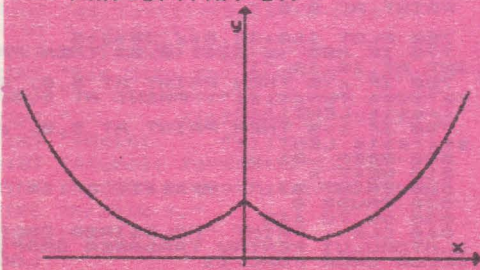
```

```

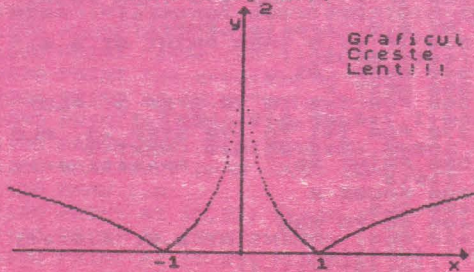
1080>NEXT x
1090 PRINT #0;" Tastati orice
!";PAUSE 0:CLS
1100 PRINT AT 2,0;"3.Graficul fu
nctiei f(x)=log |x|";AT 3,28;"2"
1110 PLOT 5,100: DRAW 245,0: DRA
W -3,3: DRAW 0,-6: DRAW 3,3: PLO
T 128,8: DRAW 0,140: DRAW 3,-3:
DRAW -6,0: DRAW 3,3
1120 PRINT AT 10,30;"x": PRINT A
T 4,15;"y"
1130 FOR x=-3 TO 3 STEP .02
1140 IF ABS x<=.07 THEN NEXT x
1150 LET y=LN (ABS x)
1160 BEEP .01,2*y+20
1170 PLOT 128+42*x,100+y*20
1180 IF ABS (x+1)<.01 THEN PRINT
AT 8,10;"-1"
1190 IF ABS (x-1)<.01 THEN PRINT
AT 8,20;"1"
1200 IF x<-1 THEN PRINT AT 11,1;
"Graficul ";AT 12,1;"Descreste"
;AT 13,1;"LENT!!"
1210 IF x>-1 AND x<0.1 THEN PRIN
T AT 13,1;"BRUSC!!"
1220 IF ABS x<.1 THEN PRINT AT 1
1,1;" ";AT 12,1;" ";AT 13,1;" "
1230 IF x>0 AND x<1 THEN PRINT A
T 12,23;"Graficul ";AT 13,23;"Cr
este ";AT 14,23;"BRUSC!!"
1240 IF x>1 THEN PRINT AT 14,23;
"LENT!!!"
1250 NEXT x
1260 PRINT #0;" Apasati orice t
asta
!";PAUSE 0:CLS
1280 PRINT AT 2,1;"4.Graficul fu
nctiei f(x)=log |x|";AT 4,17;"2"
1290 PLOT 5,9: DRAW 245,0: DRAW
-3,-3: DRAW 0,6: DRAW 3,-3: PLOT
128,8: DRAW 0,132: DRAW 3,-3: D
RAW -6,0: DRAW 3,3
1300>PRINT AT 21,30;"x": PRINT A
T 5,15;"y"
1310 FOR x=-3 TO 3 STEP 0.02
1320 IF ABS x<.07 THEN NEXT x
1330 LET y=ABS (LN (ABS x))
1340 BEEP .01,2*y+4
1350 PLOT 128+42*x,9+30*y
1360 IF x<-1 THEN PRINT AT 6,1;"
Graficul ";AT 7,1;"Descreste";AT
8,1;"LENT!!"
1370 IF ABS (x+1)<.1 THEN PRINT
AT 21,10;"-1"
1380 IF x>-1 AND x<0 THEN PRINT
AT 8,1;"BRUSC!!";PRINT AT 7,1;"
Creste"
1390 IF x>0 AND x<1 THEN PRINT A
T 6,1;" ";AT 7,1;" ";AT 8,1;" "
1400 IF x>0 AND x<1 THEN PRINT A
T 5,23;"Graficul ";AT 7,23;"Descr
este ";AT 8,23;"BRUSC!!"
1410 IF ABS (x-1)<.01 THEN PRINT
AT 21,21;"1"
1420 IF x>1 THEN PRINT AT 7,23;"
Creste ";AT 8,23;"LENT!!!"
1430 NEXT x
1440 STOP

```

6. Graficul functiei:  
 $f(x)=3f(1+|x|-1)$



4. Graficul functiei:  
 $f(x)=(\log |x|)^2$





## 18.25. Program „V. M. STAND”

1°. Tema: Calcularea valorii medii și a abaterii standard

2°. Schemă logică

3°. Program BASIC (H.C-85)

```

2 CLS
5 PRINT "CALCULUL VALORII MEDII ȘI A
  ABATERII STANDARD"
15 PRINT "CÎTE MĂSURĂTORI AȚI EFECTUAT?"
20 INPUT N
22 IF N <= 1 THEN GO TO 20
25 PRINT "DAȚI VALORILE MĂSURĂTORILOR" :
  DIM M(N)
30 FOR I=1 TO N: INPUT M(I) : NEXT I
35 LET S=0
40 FOR X=1 TO N
50 LET S=S+M (X)
55 NEXT X
60 LET V=S/N
65 LET S=0
70 FOR X=1 TO N
80 LET S=S+(M(X)-V) * (M(X)-V)
85 NEXT X
90 LET Z=SQR (S/(N-1))
95 PRINT "VALOAREA MEDIE :";V
100 PRINT "ABATEREA STANDARD : " ; Z
110 STOP
  
```

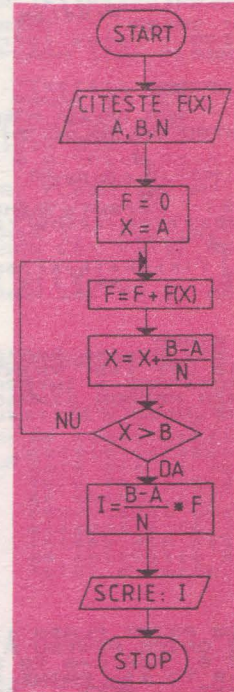


Fig. 18.2. Schema logică Dreptunghi

## 18.26. Program „Dreptunghi”

1°. Tema: Metoda dreptunghiurilor pentru calcularea integralei:

$$\int_a^b f(x) dx$$

```

60 LET F=F+VAL F $
70 NEXT X
80 LET I=((B-A)/N)*F
90 PRINT "INTEGRALA ESTE I="; I
100 STOP
  
```

Observație: Se folosește formula:

$$\int_a^b f(x) dx = \frac{b-a}{n} \sum_{i=0}^n f(x_i), \quad x_i = \frac{b-a}{n} \cdot i + a$$

2°. Schema bloc:

3°. Program BASIC (H. C.-85)

```

10 INPUT "F (X) ="; LINE F $
20 INPUT "CAPETELE INTERVALULUI:"; A; B
30 INPUT "PASUL DE DIVIZIUNE"; N
40 LET F=0
50 FOR X=A TO B STEP (B-A)/N
  
```

## 18. PROGRAME CLASELE IX—XII



## 18.27. Program „Simpson”

Tema: Metoda Simpson pentru calcul integralei definite

$$\int_a^b f(x) dx = \frac{b-a}{6p} \left[ f(a) + f(b) + 4 \sum_{i=1}^p f(x_{2i-1}) + 2 \sum_{i=1}^p f(x_{2i-2}) \right]$$

1>REM

Program „SIMPSON”

2 REM

1'. Tema: METODA SIMPSON  
PENTRU CALCULUL INTEGRALEI

$\int f(x) dx$

Formula SIMPSON (pasul de  
impartire fiind '2p')

$$\int_a^b f(x) dx = \frac{(b-a)}{6p} \left[ f(a) + f(b) + 4 \sum_{i=1}^p f(x_{2i-1}) + 2 \sum_{i=1}^p f(x_{2i-2}) \right]$$

3>REM

2'. Program BASIC (HC-85)  
10 LET f=0: PRINT "PROGRAM DE  
APROXIMARE A INTEGRALEI DEFINITE  
PRIN METODA SIMPSON"  
20 INPUT "Introduceti functia:

```

"; LINE f$
30 INPUT "Introduceti capetele
A,B": a,b
40 INPUT "Introduceti nr. inte
rvalurilor de subimpartire 2p:": p
50 LET P=P/2
60 FOR i=1 TO P
65 LET x=a+(2*i-1)*(b-a)/(2*p)
70 LET h=VAL f$
80 LET f=f+4*h
90 NEXT i
95 LET x=a
100 FOR i=2 TO P
110 LET x=a+(2*i-2)*(b-a)/(2*p)
120 LET h=VAL f$
130 LET f=f+2*h
140>NEXT i
150 LET x=a: LET h=VAL f$: LET
f=f+h
160 LET x=b: LET h=VAL f$: LET
f=f+h
170 LET i=((b-a)/(6*p))*f
180 PRINT "Valoarea integralei
este I=";i
190 STOP

```

## 18.28. Program „Compunere”

Tema: Compunerea forțelor

1 REM  
2'. Program 'COMPUNERE'

2 REM

1'. Tema: COMPUNEREA FORTE  
LOR

3 REM

2'. Program BASIC (HC-85)

```

290 PRINT FLASH 1;AT 21,0;"TAST
ATI ORICE PENTRU A CONTINUA": PA
USE 0: FOR L=0 TO 28 STEP 4: FOR
K=0 TO 21: PRINT AT K,L;" "
NEXT K: NEXT L: CLS
300 PRINT AT 0,5;"REGULA TRIUNG
HIULUI"
310 PLOT 30,56: DRAW 56,0: DRAW
-5,2: DRAW 5,-2: DRAW -5,-2: PL
OT 30,56: DRAW 40,64: DRAW -3,-3
: DRAW 3,3: DRAW 0,-4
320 PLOT 70,120: DRAW 56,0: DRA
W -5,2: DRAW 5,-2: DRAW -5,-2:
330 INK 2: PLOT 30,56: DRAW 92,
61: DRAW -2,-5: DRAW 2,5: DRAW -
6,0: INK 0
340 PRINT AT 10,3;"F1": PRINT A
T 16,6;"F2"
350 INK 2: PRINT AT 10,9;"F": I
NK 0
360 PRINT AT 9,3;"_": PRINT AT
15,6;" "
370 INK 2: PRINT AT 9,9;"_": IN
K 0
380 PRINT AT 15,2;"0": PRINT AT
15,11;"B": PRINT AT 6,7;"A": PR
INT AT 5,15;"C": PRINT AT 5,11;"
F2": PRINT AT 4,11;" "
390 PRINT OVER 1;AT -4,0;"
Se reprezinta
vectorul F1,
apoi vectorul
F2 cu punctul
de aplicatie
in virful lui
F1;rezultanta
se obtine
unind punctul
de aplicatie
al lui F1 cu
virful lui F2."
400 PRINT FLASH 1;AT 21,0;"TAST
ATI ORICE PENTRU A CONTINUA": PA
USE 0: FOR L=0 TO 28 STEP 4: FOR
K=0 TO 21: PRINT AT K,L;" "
NEXT K: NEXT L: CLS
410 PRINT FLASH 1;AT 0,10;"APLI
CATII"
420 PRINT AT 2,0;"DATI VALORILE
FORTELOR CE SE COMPUN (in newt
on)": INPUT " F1=";F1;"N, F2="
;F2;"N"
430 PRINT AT 4,0;"F1=";F1;"N"
440 PRINT AT 5,0;"F2=";F2;"N"
450 PRINT AT 6,0;"DATI MASURA U
NGHIULUI DINTRE F1 SI F2 (in ra
diani)"

```

```

10 BORDER 5: PAPER 5: INK 0: C
LS
20 LET E$="PROGRAM DE COMPUNER
E A FORTELOR"
30 FOR I=1 TO LEN E$
40 PRINT E$(I);: BEEP 0.03,10
50 NEXT I
60 LET R$=" Pentru a putea vo
rbi despre compunerea a doua for
te,trebuie sa ne reamintim ca fo
rtele cu acelasi punct de aplica
tie se numesc FORTE CONCURRENT.E
forta F care produce acelasi ef
ect in locul a doua forte F1 si
F2 se numeste FORTA REZULTANTA.R
ealizind experimentul din manual
ul dvs. se ajunge la urmatoarea
concluzie:"
70 FOR i=1 TO LEN R$
80 PRINT R$(i);:
90 NEXT I
100 LET M$=" Rezultanta a do
ua forte concurente depinde de v
alorile numerice ale forțelor,ci
t si de unghiul dintre directiil
e lor."
110 FOR i=1 TO LEN M$
120 PRINT M$(i);:
130 NEXT I
140 PRINT FLASH 1;AT 21,0;"TAST
ATI ORICE PENTRU A CONTINUA": PA
USE 0: FOR L=0 TO 28 STEP 4: FOR
K=0 TO 21: PRINT AT K,L;" "
NEXT K: NEXT L: CLS
150 PRINT AT 0,5;"REGULA PARALE
LOGRAMULUI"
160 PLOT 30,56: DRAW 56,0: DRAW
-5,2: DRAW 5,-2: DRAW -5,-2: PL
OT 30,56: DRAW 40,64: DRAW -3,-3
: DRAW 3,3: DRAW 0,-4
170 FOR J=0 TO 45 STEP 2
180 PLOT 86+J,56+1.5*J: NEXT J
190 FOR U=0 TO 70 STEP 5
200 PLOT 70+U,120: NEXT U

```



```

210 INK 2: PLOT 30,56: DRAW 10
,65: DRAW -2,-5: DRAW 2,5: DRAW
-5,-2: INK 0
220 PRINT AT 15,2:"0": PRINT AT
15,11:"B": PRINT AT 6,7:"A": PR
INT AT 5,15:"C"
230 PRINT AT 10,3:"F1": PRINT A
T 16,6:"F2"
240 INK 2: PRINT AT 10,9:"F": I
NK 0
250 PRINT AT 9,3:"_": PRINT AT
15,6:"_"
260 INK 2: PRINT AT 9,9:"_": IN
K 0
270 PRINT OVER 1: AT 4,0:"
F1+F2=F": PRINT
AT 3,22:" ": PRINT AT 3,25:"_":
PRINT AT 3,28:"_"
280 PRINT OVER 1: AT 5,0:"
Se construiesc
paralelogramul
care are ca la-
turi fortele ce
se compun, iar
rezultanta este
diagonala ce
incepe din pct.
de aplicatie al
celor 2 forte."

```

```

600 LET P=1
610 IF F1<0 THEN LET J=-1: GO T
O 630
620 LET J=1
630 IF F2<0 THEN LET K=-1: GO T
O 650
640 LET K=1
650 IF ABS (F1)>126 OR ABS (F2*CO
S U)>126 OR ABS (R*U)>126 THEN G
O TO 710
660 IF ABS (F2*SIN U)>86 OR ABS
(R*U)>86 THEN GO TO 680
670 LET Q2=1: GO TO 780
680 IF ABS (F2*SIN U)>ABS (R*U)
THEN GO TO 700
690 LET Q2=INT (ABS (R*U)/86+1)
GO TO 780
700 LET Q2=INT (ABS (F2*SIN U)/
86+1): GO TO 780
710 IF ABS (F1)>ABS (F2*COS U)
THEN GO TO 740
720 IF ABS (F2*COS U)>ABS (R*U)
THEN LET Q1=INT (ABS (F2*COS U)
/126+1): GO TO 760
730 LET Q1=INT (ABS (R*U)/126+1)
GO TO 760
740 IF ABS (F1)>ABS (R*U) THEN
LET Q1=INT (F1/126+1): GO TO 760
750 GO TO 730
760 IF (ABS (R*U)/Q1)>86 THEN L
ET L=ABS ((R*U)/Q1): LET Q2=Q1*I
NT L/86+1: GO TO 780
770 LET Q2=Q1
780 PLOT 125,86: DRAW F1/Q2-.5,
0
790 PLOT 125,86: DRAW F2*COS U/
Q2-1,F2*SIN U/Q2: FOR I=0 TO ABS
F1/Q2 STEP 4: PLOT 125+F2*COS U
/Q2+J*1,86+F2*SIN U/Q2: NEXT I
800 FOR I=0 TO ABS (F2*SIN U)/Q
2 STEP 4: PLOT 125+F1/Q2+Y*P*K*A
BS I/TAN (U-.001),86+Y*K*P*ABS
I: A XT I
810 PLOT 125,86: DRAW R*U/Q2,R*
U/Q2
820 PLOT 126,86: DRAW R*U/Q2,R*
U/Q2
830 FOR I=0 TO U STEP .1: PLOT
125+20*COS I,86+20*SIN I: NEXT I
840 LET Q=ASN W: IF Q<0 THEN LE
T Q=2*PI+Q
850 INK 2: FOR I=0 TO 0 STEP .1
: PLOT 125+10*COS I,86+10*SIN I:
NEXT I: INK 0
860 BEEP .1,30: PAUSE 0: CLS
870 PRINT AT 10,1:"MAI DORITI O
NUA APLICATIE?" (D
/N)"
880 PAUSE 0
890 IF INKEY$="D" OR INKEY$="d"
THEN CLS : GO TO 410
900 CLS : PRINT AT 10,8: FLASH
1: PAPER 6: INK 1:"LA REVEDERE"

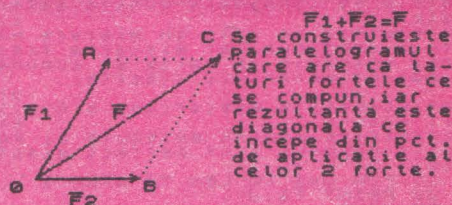
```

```

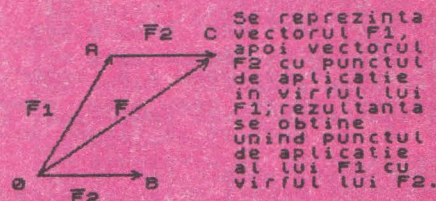
450 INPUT "U=":U
470 PRINT AT 8,0:"u=":U:" radia
ni": PRINT AT 9,0:"Rezultanta R
se calculeaza conform relatiei:"
480 PRINT AT 12,1:"R=SQR (F1+2+
F2+2*F1*F2*COS U)"
490 PAUSE 150
500 LET R=SQR (F1*F1+F2*F2+2*F1
*F2*COS U): PRINT "": PRINT "Ual
oarea rezultantei este:" PRINT
": FLASH 1:R: FLASH 1:
"N"
510 PRINT #1:"Tastati orice:"
FLASH 1:" ": PAUSE 0: CLS
520 IF R=0 THEN GO TO 560
530 LET U=(F1+F2*COS U)/R
540 LET U=F2*SIN U/R
550 PRINT AT 0,0:"F1=":F1:" N:
F2=":F2:" N:" AT 1,0: FLASH 1:"
R=": FLASH 1:R: FLASH 1:" N"
560 PRINT AT 2,0:"u=":U:" radia
ni": AT 3,0: FLASH 1:"r=":FLAS
H 1:ASN W: FLASH 1:" radiani"
570 IF W=0 THEN LET Y=-1: GO TO
590
580 LET Y=1
590 IF U<0 THEN LET P=-1: GO TO

```

### REGULA PARALELOGRAMULUI



### REGULA TRIUNGHILUI





## 18.29. Program „Teoria cinetico-moleculară”

```

1 SAVE "T.C.M.PROG" LINE 5
5 PAPER 7: INK 0: BORDER 7: C
LS : FOR F=0 TO 10: BEEP 1.4: N
EXT F
10 LET E$="
ACEST PROGRAM SE REFERA LA STU-
DIUL TEORIEI CINETICO-MOLECULARE
11 FOR J=1 TO LEN E$
12 PRINT E$(J): BEEP 0.05,10
13 NEXT J
14 PAUSE 100
20 LET R$=" Cum studiul se va
face asupra modelului de gaz id
eal, vom aminti caracteristicile
principale ale acestui model: "
21 FOR U=1 TO LEN R$ STEP 1
22 PRINT R$(U):
23 NEXT U
24 PAUSE 100
30 LET T$=" a)
Gazul este alcatuit din molecule.
b)
Datorita dimensiunilor mici, mole-
culele se considera puncte mater-
iale.
c)
Moleculele se afla in miscare ha-
otica continua, iar miscarea fiec-
arei molecule in parte se supune
legilor mecanicii clasice.
d)
Traiectoriile miscarilor sint li-
nii drepte.
e)
Ciocnirile dintre molecule si pe-
retii vasului sint perfect elast-
ice.
31 FOR J=1 TO LEN T$ STEP 1
32 PRINT T$(J):
33 NEXT J
TATI ORICE PENTRU A CONTINUA": P
AUSE 0: FOR L=0 TO 28 STEP 4: FO
R K=0 TO 21: PRINT AT K,L: " "
: NEXT K: NEXT L: CLS
40 PRINT "CIOCNIREA UNEI MOLEC
ULE CU UN PERETE
45 FOR I=1 TO 5
47 PLOT 24+I,120: DRAW 0,-73
48 NEXT I
50 PLOT 5,83: DRAW 100,0
55 PLOT 5,83: DRAW 4,2: DRAW -
4,-2: DRAW 4,-2
60 PLOT 29,78: DRAW 0,60: PLOT
29,138: DRAW 2,-4: DRAW -2,4: D
RAW -2,-4: PLOT 25,45: DRAW -5,-
10
65 PRINT AT 18,0: "perete"
70 FOR I=1 TO 6
80 PRINT OVER 1, AT 5+I,9-I: "♦"
: PAUSE 25: PRINT OVER 1, AT 5+I,
9-I: "♦"
90 NEXT I
99 FOR I=0 TO 4: PLOT 25+I,70:
DRAW 0,20: NEXT I
100 FOR I=1 TO 6
109 PLOT 5,83: DRAW 70,0
110 PRINT AT 12+I,5+I: "♦": PAUS
E 25: PRINT OVER 1, AT 12+I,5+I: "
♦"
111 PLOT 5,83: DRAW 70,0
112 NEXT I
1000 LET L=US: "o"
1010 POKE L+0,BIN 00011000
1020 POKE L+1,BIN 00111100
1030 POKE L+2,BIN 01111110
1040 POKE L+3,BIN 01111110
1050 POKE L+4,BIN 01111110
1060 POKE L+5,BIN 00011100
1070 POKE L+6,BIN 00011000
1080 POKE L+7,BIN 00011000
1100 FOR U=0 TO 50 STEP 5: PLOT
25+U,83+U: DRAW 0: NEXT U
1200 FOR U=0 TO 50 STEP 5: PLOT
25+U,83+U: DRAW 3,-3: NEXT U
1510 PLOT 79,193: DRAW -15,-15:
PLOT 79,193: DRAW -15: PLOT 79
,193: DRAW -15,0: DRAW 3,1: DRAW
-3,-1: DRAW 3,-1: DRAW -3,1: PL
OT 79,118: DRAW 1,3: DRAW -1,-3:
DRAW -1,3: DRAW 1,-3: PLOT 64,1
18: DRAW 0,2: DRAW 0,-2: DRAW 2,
0
1220 PLOT 64,48: DRAW 15,-15: DR
AW -2,0: DRAW 2,0: DRAW 0,2: PLO
T 64,48: DRAW 15,0: DRAW 2,1: D
RAW 2,-1: DRAW -2,-1: PLOT 64,48
: DRAW 0,-15: DRAW 1,2: DRAW -1,
2: DRAW -1,2
1230 FOR U=0 TO 100 STEP 4: PLOT
64,48+U: NEXT U
1235 FOR U=0 TO 100 STEP 4: PLOT
79,33+U: NEXT U
1240 PRINT AT 4,4: "X"
1250 FOR U=0 TO 40 STEP 4: PLOT
64+U,118: PLOT 64+U,33: NEXT U
1260 PRINT AT 12,0: "Z"
1270 PRINT AT 4,6: "U": PRINT A
T 6,10: "-UX" PRINT AT 7,7: "U"
1280 PRINT AT 14,9: "-UZ" PRINT
AT 17,5: "-UX": PRINT AT 18,9: "U"
1300 PRINT AT 4,0: OVER 1,
U=viteza molecu-
lei
UZ,UX=proiectii-
le lui U pe axe-
le OX si OZ"
1350 PRINT AT 9,0: OVER 1,
Presiunea exer-
citata de gaz a-
supra peretilor
vasului este e-
gala cu valoarea
componentelor a-
cesteia pe axe-
le de coordonate, si
este data de for-
mula

$$P = \frac{2}{3} n \cdot m \cdot U^2$$

1360 PRINT AT 21,0: FLASH 1: "TAS
TATI ORICE PENTRU A CONTINUA": P
AUSE 0: FOR L=0 TO 28 STEP 4: FO
R K=0 TO 21: PRINT AT K,L: " "
: NEXT K: NEXT L: CLS
1370 PRINT "Privind tridimension-
al ciocnirea, avem urmatoarea rep-
rezentare in trei axe de coordon-
ate.
1380 PLOT 10,35: DRAW 80,0: DRAW
35,35: DRAW -80,0: DRAW -35,35
DRAW 0,-5: DRAW 80,0: DRAW 0,5
DRAW 0,-5: DRAW 35,35: DRAW 0,
5: PLOT 55,30: DRAW -10,-7: PRIN
T AT 10,3: "perete"
1390 PLOT 50,57: DRAW 0,45: PLOT
50,57: DRAW 40,0: PLOT 50,57: D
RAW -15,-15
1400 PLOT 50,102: DRAW -2,-5: DR
AW 2,5: DRAW 2,-5: PLOT 90,57: D
RAW -5,-2: DRAW 5,2: DRAW -5,2:
PLOT 35,42: DRAW 5,2: DRAW -5,-2
: DRAW 2,5
1410 INK 5: PLOT 50,104: DRAW 0,
20: PLOT 92,57: DRAW 41,0: PLOT
34,40: DRAW -20,-20: DRAW 5,2: D
RAW -5,-2: DRAW 2,5: PLOT 133,57
: DRAW -5,-2: DRAW 5,2: DRAW -5,
2: PLOT 50,124: DRAW -2,-5: DRAW
2,5: DRAW 2,-5: INK 0
1420 INK 3: PRINT AT 6,5: "Z": PR
INT AT 20,1: "X": PRINT AT 15,16:
"Y"
1430 FOR U=0 TO 50 STEP 4: PLOT
35+U,41: FOR U=0 TO 20 STEP 3: P
LOT 92,57+U: NEXT U: NEXT U
1450 PLOT 50,57: DRAW 26,-15: DR
AW -5,0: DRAW 5,0: DRAW -2,5
1460 FOR U=0 TO 30 STEP 3: PLOT
50+U,102-0.5*U: NEXT U
1470 FOR U=0 TO 50 STEP 3: PLOT
76,42+U: NEXT U
1480 PLOT 50,57: DRAW 27,32: DRA
W -1,-5: DRAW 1,5: DRAW -5,-2: P
LOT 43,50: DRAW -10,20
1490 PRINT OVER 1, AT 12,3: "UX":
PRINT AT 11,12: "UY": PRINT AT 11
,7: "U"

```



```

1500 PLOT 85,57: DRAW 10,20: PRI
NT AT 10,4;"0";
1510 PRINT OVER 1;AT 9,4;" ": PR
INT OVER 1;AT 11,3;" ": PRINT OU
ER 1;AT 10,12;"-": PRINT OVER 1;
AT 10,7;"-";
1520 LET T=USR "E"
1530 POKE L+1,BIN 00011000
1540 POKE L+2,BIN 00100100
1550 POKE L+3,BIN 00100000
1560 POKE L+4,BIN 00100000
1570 POKE L+5,BIN 00100000
1580 POKE L+6,BIN 00100100
1590 POKE L+7,BIN 00011000
1610 PRINT OVER 1;AT 4,0;"
Deci:
P=(2/3)*n*(m*
v^2),unde
n=nr.molecule
m=masa unei
molecule
v=viteza unei
molecule"
1620 PRINT OVER 1;AT 12,0;"
Raportul"
1630 INK 2: PRINT OVER 1;AT 13,0
;"
E=(m*v^2)/2"
1635 PRINT OVER 1;AT 14,0;"
reprezinta ener
gia cinetica
medie a unei
molecule"
1640 PRINT AT 21,0: FLASH 1;"TAS
TATI ORICE PENTRU A CONTINUA": P
AUSE 0: FOR L=0 TO 28 STEP 4: FO

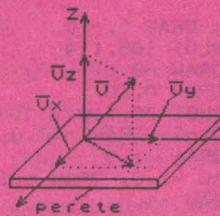
```

```

R k=0 TO 21: PRINT AT k,l;"
: NEXT k: NEXT l: CLS
1650 PRINT AT 1,10;"CONCLUZIE"
1655 INK 2: PRINT AT 4,0;"PRESIU
NEA GAZULUI ESTE NUMERIC EGALA C
U DOUA TREIMI DIN ENERGIA CINETI
CA MEDIE A TUTUROR MOLECULELOR D
E GAZ CUPRINSE IN UNITATEA DE VO
LUM": INK 0
1657 PRINT AT 12,10;"P=(2/3)*n*(
m*v^2)/2"

```

Privind tridimensional ciocnirea, avem următoarea reprezentare în trei axe de coordonate:



```

Deci:
P=(2/3)*n*(m*
v^2),unde
n=nr.molecule
m=masa unei
molecule
v=viteza unei
molecule
Raportul
E=(m*v^2)/2
reprezinta ener
gia cinetica
medie a unei
molecule

```

TASTATI ORICE PENTRU A CONTINUA

## 18.30. Program „Lentila”

**Tema:** Studiul imaginilor formate de lentile subțiri

```

1 REM
2. Program "LENTILA"
2 REM
1' Tema: Studiul imaginilor
lor formate de lentile subțiri
3 REM
2' Program BASIC (HC-85)
10 CLS
20 PRINT "Studiul imaginilor f
ormate de lentile subțiri"
30 PRINT
40 PRINT "Introduceti distanta
focala"
50 PRINT "Pozitiva pentru lent
ita convergenta"
60 PRINT "Negativa pentru lent
ita divergenta"
70 INPUT "F= ";f

```

```

80 IF F=0 THEN GO TO 70
90 PRINT "Lentila este ";
100 IF f>0 THEN PRINT "CONVERGE
NTA"
110 IF f<0 THEN PRINT "DIVERGEN
TA"
120 PRINT "Distanța focala F="
130 PRINT "Coordonatele punctul
ui obiect"
140 PRINT "X= ";
150 INPUT X
160 IF ABS X=ABS F THEN PRINT "
Obiectul este situat în planul
focal"
170 IF X=0 THEN GO TO 150
180 PRINT X
190 PRINT "Y= ";
200 INPUT Y
210 PRINT Y
220 PRINT
230 LET X1=(X*F)/(X+f)
240 LET Y1=(Y*F)/(X+f)
250 PRINT "Coordonatele punctul
ui imagine"
260 PRINT "X'= ";Y1
270 PRINT "Y'= ";Y1
280 PRINT
290 PRINT "Apasa orice tasta..."
300 PAUSE 0: GO TO 10

```



## 18.31. Program „Optica”

```

1 REM 31. Program "OPTICA"
2 REM 1'.Tema: Probleme de optica
3 REM 2'.Program BASIC (HC-85)
10 BORDER 2: PAPER 2: INK 0: CLS
20 PRINT "Programul de fata studiază"
30 PRINT "formarea imaginilor in "
40 PRINT "aparatul de proiectie"
50 PAUSE 0: CLS
60 PLOT 20,75: DRAW -10 ,10,-2: DRAW 0,70
70 DRAW 10,10,-2: DRAW 5,0
80 DRAW 35,0: DRAW 10,-10,-2: DRAW 0,-25
90 DRAW 12,0: DRAW 0,-2: DRAW 8,0
100 DRAW 0,-2: DRAW 15,0
110 PLOT 100,125: DRAW 25,0: DRAW 0,-10
120 DRAW -25,0: DRAW 0,10: PLOT 105,114
130 DRAW -15,0: DRAW 0,-2: DRAW -8,0
140 DRAW 0,-2: DRAW -15,0: DRAW 0,20
150 DRAW 15,0: DRAW 0,-20: PLOT 70,110
160 DRAW 0,-25: DRAW -10,-10,-2: DRAW -40,0
170 PLOT 43,75: DRAW 0,17: DRAW -1,0
180 DRAW -1,3: DRAW 0,8: DRAW -1,0: DRAW 0,2
190 DRAW 10,0: DRAW 0,-2: DRAW -1,0
200 DRAW 0,-8: DRAW -1,-3: DRAW -1,0
210 DRAW 0,-17: PLOT 43,92: DRAW 4,0
220 PLOT 41,95: DRAW 8,0: PLOT 41,103
230 DRAW 8,0
240 PLOT 43,105: DRAW 0,5: DRAW -3,10
250 FOR I=1 TO 2: DRAW 5,0,-3
260 NEXT I
270 DRAW -3,-10: DRAW 0,-5
280 PLOT 41,105: DRAW -6,10,1
290 DRAW 10,15,-2.3: DRAW 10,-15,-2.3
300 DRAW -6,-10,1
310 PLOT 33,105: DRAW 0,30,-2.5
320 PLOT 80,110: DRAW 0,20: DRAW 0,-20,2
330 PLOT 68,110: DRAW 0,20: DRAW 0,-20,-2
340 CIRCLE 87,103,2: PLOT 85,103
350 DRAW 0,10: DRAW -2,2: DRAW 1,0
360 DRAW 0,11: DRAW 2,0: DRAW 0,-11
370 DRAW 1,0: DRAW -2,-2: PLOT 85,125
380 DRAW 0,10: CIRCLE 87,138,2
390 PLOT 187,80: DRAW 0,90: DRAW 40,-10
400 DRAW 0,-90: DRAW -40,10
410 GO TO 530
420 FOR I=1 TO 20: PLOT 45+I,120+I*.5
430 PLOT 45+I,120-I*.5
440 NEXT I
450 FOR I=1 TO 140: PLOT 65+I,130-I*(27/140)
460 PLOT 65+I,110+I*(27/140)
470 NEXT I
480 PAUSE 20
490 PLOT 207,103: DRAW 0,31: DRAW 2,0
500 DRAW -3,2: DRAW -3,-2: DRAW 2,0
510 DRAW 0,-31: DRAW 3,0
520 GO SUB 1480: GO TO 920
530 PRINT AT 3,4;"2";AT 3,6;"1"
540 PRINT AT 3,9;"3";AT 3,13;"4"
550 PRINT AT 10,13;"5"
560 PLOT 45,130: DRAW 5,15: PLOT 33,134
570 DRAW 3,8: PLOT 75,127: DRAW 0,15
580 PLOT 110,125
590 DRAW -3,20: PLOT 87,114
600 DRAW 20,-20
610 PRINT AT 14,0;
620 PRINT "Aparatul de proiectie este "
630 PRINT "format din urmatoarele parti:"
640 PAUSE 0: PRINT AT 14,0;
650 GO SUB 1480
660 PRINT AT 14,0;
670 PRINT "1) Bec cu incandescenta "
680 PRINT "2) Oglinza concava";
685 PRINT " "
690 PRINT "3) Lentile condensatoare"
700 PRINT "4) Obiectiv"
710 PRINT "5) Obiect"
720 PAUSE 0
730 PRINT AT 3,4;" ";AT 3,6;" ";AT 3,9;" "
740 PRINT AT 3,13;" ";AT 10,13;" "
750 PLOT INVERSE 1;45,130
760 DRAW OVER 1;5,15
770 PLOT INVERSE 1;33,134
780 DRAW OVER 1;3,7
790 PLOT INVERSE 1;75,127
800 DRAW OVER 1;0,15
810 PLOT INVERSE 1;110,125
820 DRAW OVER 1;-3,20
830 PLOT INVERSE 1;87,114
840 DRAW OVER 1;20,-20
850 PRINT AT 14,0;
860 PRINT "Razele de lumina concentrate de"
870 PRINT "oglinza concava sint proiectate"
880 PRINT "pe obiect prin grupul de lentile"
890 PRINT "ce alcatuiesc CONDENSATORUL,apoi"
900 PRINT "pe ecran prin OBIECTIV."
910 GO TO 420
920 PAUSE 0: CLS : PLOT 10,140: DRAW 200,0
930 PLOT 20,140: DRAW 0,5: DRAW -2,-2
940 DRAW 4,0: DRAW -2,2: PLOT 40,125
950 DRAW 0,30: PLOT 38,155
960 DRAW 2,5: DRAW 2,-5: DRAW -4,0
970 PLOT 138,110: DRAW 0,60
980 PRINT AT 0,5;"O(1)"
990 PRINT AT 0,16;"O(2)"
1000 PRINT AT 14,0;
1010 PRINT "Microscopul optic este format";
1020 PRINT " "
1030 PRINT "obiectivul O(1) si ocularul O(2)"
1040 GO SUB 1480
1045 PAUSE 0
1050 PRINT AT 12,0;
1060 PRINT "Obiectul foarte mic,este";
1070 PRINT " plasat "
1080 PRINT "in focarul primei lentile";
1090 PRINT " ,iar "
1100 PRINT " imaginea se formeaza ";
1110 PRINT " RASTURNATA,";
1120 PRINT "MARITA si REALA "
1130 PRINT "Aceasta imagine reprezinta";
1135 PRINT " "
1140 PRINT "obiectul pentru cea ";
1150 PRINT "de-a doua "
1160 PRINT "lentila care formeaza";
1170 PRINT " imaginea"
1180 PRINT "RASTURNATA,VIRTUALA si ";
1190 PRINT "MULT MARITA"
1200 PLOT 10,140: DRAW 200,0: PLOT 20,140
1210 DRAW 0,5: DRAW -2,-2: DRAW 2,2

```



```

1220 DRAW 2,-2: PLOT 40,125: DRAW 0,30
1230 PLOT 40,125: DRAW -2,5: DRAW 4,0
1240 DRAW -2,-5
1250 PLOT 138,110
1260 DRAW 0,60: PLOT 136,165
1270 DRAW 2,5: DRAW 2,-5: PLOT 136,115
1280 DRAW 2,-5: DRAW 2,5
1290 FOR q=1 TO 95
1300 PLOT 20+q,145+q*(-.176)
1310 NEXT q
1320 FOR q=1 TO 20
1330 PLOT 20+q,145+q*(-.5)
1340 NEXT q
1350 FOR q=1 TO 75
1360 PLOT 40+q,135+q*(-.08)
1370 NEXT q
1380 PLOT 115,140: DRAW 0,-10: DRAW -2 ,3
1390 DRAW 4,0: DRAW -2,-3
1400 CIRCLE 185,140,1: PRINT AT 5,24;"F(1)"
1410 CIRCLE 91,140,1: PRINT AT 3,11;"F'(1)"
1420 FOR q=1 TO 140: PLOT 160-q,150+q*(-.492)
1430 NEXT q
1440 FOR q=1 TO 180: PLOT 200-q,145-q*(-.35)
1450 NEXT q
1460 PLOT 20,140: DRAW 0,-60: DRAW -2,5
1470 DRAW 4,0: DRAW -2,-5
1475 STOP
1480 FOR m=12 TO 21
1490 PAUSE 2
1500 PRINT AT m,0'
1510 NEXT m
1520 RETURN

```

## 18.32. Program „Motoare termice”

```

1 BORDER 0: INK 7: PAPER 0: C
LS
155 CLS: PRINT AT 21,0;" "
156 POKE 23692,255
157 LET X$=" TIPURI DE MOTOARE
TERMICE

```

```

N ARDEREA IN MOTOR A UNUI COMBUS
TIBIL ESTE TRANSMISA SUBSTANTE
I DE LUCRU, CARE ISI MARESTE PR
ESIUNEA SI APASA PE PISTONUL M
OBIL AL UNUI CILINDRU, PUNINDU-L
IN MISCARE

```

```

IN FUNCTIE DE CONST
MOTORULUI, COMBUSTI
BILUL ARDE INEXTERIORUL SAU IN I
CILINDRULUI.

```

```

MOTOARELE SE IMPART
IN 2 GRUPE: -TERMICE CU ARDERE
EXTERNA (MOTORUL CU ABUR

```

```

I, TURBINA) -TERMICE CU ARDERE
INTERNA (MOTORUL DIESEL,
OTTO, CU REACTIE)

```

```

160 FOR F=1 TO LEN X$ STEP 4
162 PAUSE 2: PRINT X$(F);X$(F+1
);X$(F+2);X$(F+3);
165 NEXT F
190 GO SUB 9100
999 PAUSE 0: CLS
1000 REM MOTOR OTTO
1100 GO SUB 9000
1105 GO SUB 9050
1106 GO TO 1110

```

```

1107 FOR f=10 TO 19: PRINT AT f,
16," " NEXT f: PRINT AT
14,16;"CICLUL":AT 16,16;"REINCEP
E": PAUSE 60: PRINT AT 14,16;"
":AT 16,16;"

```

```

1110 FOR K=0 TO 10: PLOT 31,60+K
: DRAW 36,0: NEXT K
1120 PRINT AT 21,0: INVERSE : "A
SPIRATIA": PRINT AT 0,0;"TIMPUL

```

```

-1- (ASPIRATIA) PISTONU
L COBOARA IN CILINDRU. SUPAPA
DE ADMISIE SE DESCHIDE SI AMES
TECUL DE VAPORI DE AER SI BENZ
INA ESTE ABSORBIT.

```

```

1121 CIRCLE 140,25,2: PLOT 140,2
5: DRAW 40,0: CIRCLE 180,25,2
1125 PLOT OVER 1;15,100: DRAW OV
ER 1;10,0: PLOT OVER 1;15,101: D
RAW OVER 1;10,0

```

```

1126 PLOT 15,109: DRAW 10,0: PLO
T 15,110: DRAW 10,0
1130 FOR Y=90 TO 50 STEP -1:
1135 PLOT OVER 1;31,Y: DRAW OVER
1;38,0

```

```

1136 PLOT OVER 1;31,Y-11: DRAW O
VER 1;38,0
1150 NEXT Y
1160 PLOT 180,25: DRAW -40,25,-P
I/4: CIRCLE 140,50,2: PRINT AT
21,0;"COMPRESIA ": PRINT AT 0,0
;"TIMPUL -2- (COMPRESIA)

```

```

AMBELE SUPAPE SE INCHID, IAR
PISTONUL SE MISCA SPRE PUNCTUL
P.M. SUPERIOR COMPRIMIND
AMESTECUL CARBURANT

```

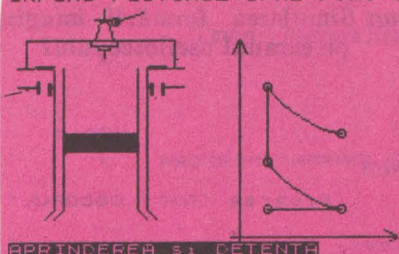


```

1165 OVER 1: PLOT 15,109: DRAW 1
0,0: PLOT 15,110: DRAW 10,0: OVE
R 0
1167 PLOT OVER 1;15,100: DRAW OV
ER 1;10,0: PLOT OVER 1;15,101: D
RAW OVER 1;10,0
1168 FOR Y=50 TO 90 STEP 1
1173 PLOT OVER 1;31,Y: DRAW OVER
1;38,0: PLOT OVER 1;31,Y-11: DR
AW OVER 1;38,0
1180 NEXT Y
1200 PLOT 140,50: DRAW 0,40: CIR
CLE 140,90,2: PRINT AT 21,0: INV
ERSE 1: "APRINDEREA SI DETENTA":
PRINT AT 0,0: "TIMPUL -3- (APRINDER
EA SI DETENTA) SE PRODUCE O SCIN
TEIE INTRA ELECTROZII BUJIEI
CARE APRINDE AMESTECUL CARBURAN
T.GAZELE INPING PISTONUL SP
RE P.M. INF."
1210 FOR A=0 TO 20: OUT 254,7: P
RINT OVER 1: BRIGHT 1: FLASH 1: A
T 9,5: "*** PRINT AT 9,5: FLASH
0,4: OUT 254,0: NEXT A: PRI
NT AT 9,5: FLASH 0: "
1211 PLOT 140,90: DRAW 40,-25:PI
4: CIRCLE 180,65,2
1220 FOR Y=90 TO 50 STEP -1: PLO
T OVER 1;31,Y: DRAW OVER 1;38,0:
PLOT OVER 1;31,Y-11: DRAW OVER
1;38,0
1230 NEXT Y
1250 PLOT 180,65: DRAW 0,-40: PR
INT AT 21,0: "EVACUAREA
4- (EVACUAREA) SUPAPA D
E MISCA DIN P.M. INF. SPRE P.M
ELIERIOR SI IMPINGE AFARA GA
ZE ARSE."
1260 OVER 1: PLOT 76,100: DRAW 9
0,0: PLOT 76,101: DRAW 9,0: OVER
1270 OVER 1: PLOT 76,109: DRAW 9
0,0: PLOT 76,110: DRAW 9,0: OVER
1280 FOR Y=50 TO 90 STEP 1: PLOT
OVER 1;31,Y: DRAW OVER 1;38,0:
PLOT OVER 1;31,Y-11: DRAW OVER 1
;38,0: NEXT Y
1290 OVER 1: PLOT 76,109: DRAW 9
0,0: PLOT 76,110: DRAW 9,0: OVER
1300 OVER 1: PLOT 76,100: DRAW 9
0,0: PLOT 76,101: DRAW 9,0: OVER
1310 GO TO 1107
1320 STOP
1330 PLOT 20,20: DRAW 5,5: DRAW
0,0: PLOT 25,20: DRAW 5,5: DRAW
0,0: PLOT 30,20: DRAW 5,5: DR
AW 0,0: PLOT 35,20: DRAW 5,5: DR
AW 0,0: PLOT 40,20: DRAW 5,5:
DRAW 0,1: PLOT 45,20: DRAW 0,1
5: PLOT 50,20: DRAW 0,15: PLOT
55,20: DRAW 0,30: PLOT 60,20:
DRAW 0,45: PLOT 65,20: DRAW 0,6
0: PLOT 70,20: DRAW 0,75: PLOT
75,20: DRAW 0,90: PLOT 80,20:
DRAW 0,100: PLOT 85,20: DRAW 0,1
15: PLOT 90,20: DRAW 0,130: PLOT
95,20: DRAW 0,150: PLOT 100,20:
DRAW 0,170: PLOT 105,20: DRAW 0,1
90: PLOT 110,20: DRAW 0,210: PLOT
115,20: DRAW 0,230: PLOT 120,20:
DRAW 0,250: PLOT 125,20: DRAW 0,2
70: PLOT 130,20: DRAW 0,290: PLOT
135,20: DRAW 0,310: PLOT 140,20:
DRAW 0,330: PLOT 145,20: DRAW 0,3
50: PLOT 150,20: DRAW 0,370: PLOT
155,20: DRAW 0,390: PLOT 160,20:
DRAW 0,410: PLOT 165,20: DRAW 0,4
30: PLOT 170,20: DRAW 0,450: PLOT
175,20: DRAW 0,470: PLOT 180,20:
DRAW 0,490: PLOT 185,20: DRAW 0,5
10: PLOT 190,20: DRAW 0,530: PLOT
195,20: DRAW 0,550: PLOT 200,20:
DRAW 0,570: PLOT 205,20: DRAW 0,5
90: PLOT 210,20: DRAW 0,610: PLOT
215,20: DRAW 0,630: PLOT 220,20:
DRAW 0,650: PLOT 225,20: DRAW 0,6
70: PLOT 230,20: DRAW 0,690: PLOT
235,20: DRAW 0,710: PLOT 240,20:
DRAW 0,730: PLOT 245,20: DRAW 0,7
50: PLOT 250,20: DRAW 0,770: PLOT
255,20: DRAW 0,790: PLOT 260,20:
DRAW 0,810: PLOT 265,20: DRAW 0,8
30: PLOT 270,20: DRAW 0,850: PLOT
275,20: DRAW 0,870: PLOT 280,20:
DRAW 0,890: PLOT 285,20: DRAW 0,9
10: PLOT 290,20: DRAW 0,930: PLOT
295,20: DRAW 0,950: PLOT 300,20:
DRAW 0,970: PLOT 305,20: DRAW 0,9
90: PLOT 310,20: DRAW 0,1010: PLOT
315,20: DRAW 0,1030: PLOT 320,20:
DRAW 0,1050: PLOT 325,20: DRAW 0,1
070: PLOT 330,20: DRAW 0,1090: PLOT
335,20: DRAW 0,1110: PLOT 340,20:
DRAW 0,1130: PLOT 345,20: DRAW 0,1
150: PLOT 350,20: DRAW 0,1170: PLOT
355,20: DRAW 0,1190: PLOT 360,20:
DRAW 0,1210: PLOT 365,20: DRAW 0,1
230: PLOT 370,20: DRAW 0,1250: PLOT
375,20: DRAW 0,1270: PLOT 380,20:
DRAW 0,1290: PLOT 385,20: DRAW 0,1
310: PLOT 390,20: DRAW 0,1350: PLOT
395,20: DRAW 0,1370: PLOT 400,20:
DRAW 0,1390: PLOT 405,20: DRAW 0,1
410: PLOT 410,20: DRAW 0,1450: PLOT
415,20: DRAW 0,1470: PLOT 420,20:
DRAW 0,1490: PLOT 425,20: DRAW 0,1
510: PLOT 430,20: DRAW 0,1550: PLOT
435,20: DRAW 0,1570: PLOT 440,20:
DRAW 0,1590: PLOT 445,20: DRAW 0,1
610: PLOT 450,20: DRAW 0,1650: PLOT
455,20: DRAW 0,1670: PLOT 460,20:
DRAW 0,1690: PLOT 465,20: DRAW 0,1
710: PLOT 470,20: DRAW 0,1750: PLOT
475,20: DRAW 0,1770: PLOT 480,20:
DRAW 0,1790: PLOT 485,20: DRAW 0,1
810: PLOT 490,20: DRAW 0,1850: PLOT
495,20: DRAW 0,1870: PLOT 500,20:
DRAW 0,1890: PLOT 505,20: DRAW 0,1
910: PLOT 510,20: DRAW 0,1950: PLOT
515,20: DRAW 0,1970: PLOT 520,20:
DRAW 0,1990: PLOT 525,20: DRAW 0,2
010: PLOT 530,20: DRAW 0,2050: PLOT
535,20: DRAW 0,2070: PLOT 540,20:
DRAW 0,2090: PLOT 545,20: DRAW 0,2
110: PLOT 550,20: DRAW 0,2150: PLOT
555,20: DRAW 0,2170: PLOT 560,20:
DRAW 0,2190: PLOT 565,20: DRAW 0,2
210: PLOT 570,20: DRAW 0,2250: PLOT
575,20: DRAW 0,2270: PLOT 580,20:
DRAW 0,2290: PLOT 585,20: DRAW 0,2
310: PLOT 590,20: DRAW 0,2350: PLOT
595,20: DRAW 0,2370: PLOT 600,20:
DRAW 0,2390: PLOT 605,20: DRAW 0,2
410: PLOT 610,20: DRAW 0,2450: PLOT
615,20: DRAW 0,2470: PLOT 620,20:
DRAW 0,2490: PLOT 625,20: DRAW 0,2
510: PLOT 630,20: DRAW 0,2550: PLOT
635,20: DRAW 0,2570: PLOT 640,20:
DRAW 0,2590: PLOT 645,20: DRAW 0,2
610: PLOT 650,20: DRAW 0,2650: PLOT
655,20: DRAW 0,2670: PLOT 660,20:
DRAW 0,2690: PLOT 665,20: DRAW 0,2
710: PLOT 670,20: DRAW 0,2750: PLOT
675,20: DRAW 0,2770: PLOT 680,20:
DRAW 0,2790: PLOT 685,20: DRAW 0,2
810: PLOT 690,20: DRAW 0,2850: PLOT
695,20: DRAW 0,2870: PLOT 700,20:
DRAW 0,2890: PLOT 705,20: DRAW 0,2
910: PLOT 710,20: DRAW 0,2950: PLOT
715,20: DRAW 0,2970: PLOT 720,20:
DRAW 0,2990: PLOT 725,20: DRAW 0,3
010: PLOT 730,20: DRAW 0,3050: PLOT
735,20: DRAW 0,3070: PLOT 740,20:
DRAW 0,3090: PLOT 745,20: DRAW 0,3
110: PLOT 750,20: DRAW 0,3150: PLOT
755,20: DRAW 0,3170: PLOT 760,20:
DRAW 0,3190: PLOT 765,20: DRAW 0,3
210: PLOT 770,20: DRAW 0,3250: PLOT
775,20: DRAW 0,3270: PLOT 780,20:
DRAW 0,3290: PLOT 785,20: DRAW 0,3
310: PLOT 790,20: DRAW 0,3350: PLOT
795,20: DRAW 0,3370: PLOT 800,20:
DRAW 0,3390: PLOT 805,20: DRAW 0,3
410: PLOT 810,20: DRAW 0,3450: PLOT
815,20: DRAW 0,3470: PLOT 820,20:
DRAW 0,3490: PLOT 825,20: DRAW 0,3
510: PLOT 830,20: DRAW 0,3550: PLOT
835,20: DRAW 0,3570: PLOT 840,20:
DRAW 0,3590: PLOT 845,20: DRAW 0,3
610: PLOT 850,20: DRAW 0,3650: PLOT
855,20: DRAW 0,3670: PLOT 860,20:
DRAW 0,3690: PLOT 865,20: DRAW 0,3
710: PLOT 870,20: DRAW 0,3750: PLOT
875,20: DRAW 0,3770: PLOT 880,20:
DRAW 0,3790: PLOT 885,20: DRAW 0,3
810: PLOT 890,20: DRAW 0,3850: PLOT
895,20: DRAW 0,3870: PLOT 900,20:
DRAW 0,3890: PLOT 905,20: DRAW 0,3
910: PLOT 910,20: DRAW 0,3950: PLOT
915,20: DRAW 0,3970: PLOT 920,20:
DRAW 0,3990: PLOT 925,20: DRAW 0,4
010: PLOT 930,20: DRAW 0,4050: PLOT
935,20: DRAW 0,4070: PLOT 940,20:
DRAW 0,4090: PLOT 945,20: DRAW 0,4
110: PLOT 950,20: DRAW 0,4150: PLOT
955,20: DRAW 0,4170: PLOT 960,20:
DRAW 0,4190: PLOT 965,20: DRAW 0,4
210: PLOT 970,20: DRAW 0,4250: PLOT
975,20: DRAW 0,4270: PLOT 980,20:
DRAW 0,4290: PLOT 985,20: DRAW 0,4
310: PLOT 990,20: DRAW 0,4350: PLOT
995,20: DRAW 0,4370: PLOT 1000,20:
DRAW 0,4390: PLOT 1005,20: DRAW 0,4
410: PLOT 1010,20: DRAW 0,4450: PLOT
1015,20: DRAW 0,4470: PLOT 1020,20:
DRAW 0,4490: PLOT 1025,20: DRAW 0,4
510: PLOT 1030,20: DRAW 0,4550: PLOT
1035,20: DRAW 0,4570: PLOT 1040,20:
DRAW 0,4590: PLOT 1045,20: DRAW 0,4
610: PLOT 1050,20: DRAW 0,4650: PLOT
1055,20: DRAW 0,4670: PLOT 1060,20:
DRAW 0,4690: PLOT 1065,20: DRAW 0,4
710: PLOT 1070,20: DRAW 0,4750: PLOT
1075,20: DRAW 0,4770: PLOT 1080,20:
DRAW 0,4790: PLOT 1085,20: DRAW 0,4
810: PLOT 1090,20: DRAW 0,4850: PLOT
1095,20: DRAW 0,4870: PLOT 1100,20:
DRAW 0,4890: PLOT 1105,20: DRAW 0,4
910: PLOT 1110,20: DRAW 0,4950: PLOT
1115,20: DRAW 0,4970: PLOT 1120,20:
DRAW 0,4990: PLOT 1125,20: DRAW 0,5
010: PLOT 1130,20: DRAW 0,5050: PLOT
1135,20: DRAW 0,5070: PLOT 1140,20:
DRAW 0,5090: PLOT 1145,20: DRAW 0,5
110: PLOT 1150,20: DRAW 0,5150: PLOT
1155,20: DRAW 0,5170: PLOT 1160,20:
DRAW 0,5190: PLOT 1165,20: DRAW 0,5
210: PLOT 1170,20: DRAW 0,5250: PLOT
1175,20: DRAW 0,5270: PLOT 1180,20:
DRAW 0,5290: PLOT 1185,20: DRAW 0,5
310: PLOT 1190,20: DRAW 0,5350: PLOT
1195,20: DRAW 0,5370: PLOT 1200,20:
DRAW 0,5390: PLOT 1205,20: DRAW 0,5
410: PLOT 1210,20: DRAW 0,5450: PLOT
1215,20: DRAW 0,5470: PLOT 1220,20:
DRAW 0,5490: PLOT 1225,20: DRAW 0,5
510: PLOT 1230,20: DRAW 0,5550: PLOT
1235,20: DRAW 0,5570: PLOT 1240,20:
DRAW 0,5590: PLOT 1245,20: DRAW 0,5
610: PLOT 1250,20: DRAW 0,5650: PLOT
1255,20: DRAW 0,5670: PLOT 1260,20:
DRAW 0,5690: PLOT 1265,20: DRAW 0,5
710: PLOT 1270,20: DRAW 0,5750: PLOT
1275,20: DRAW 0,5770: PLOT 1280,20:
DRAW 0,5790: PLOT 1285,20: DRAW 0,5
810: PLOT 1290,20: DRAW 0,5850: PLOT
1295,20: DRAW 0,5870: PLOT 1300,20:
DRAW 0,5890: PLOT 1305,20: DRAW 0,5
910: PLOT 1310,20: DRAW 0,5950: PLOT
1315,20: DRAW 0,5970: PLOT 1320,20:
DRAW 0,5990: PLOT 1325,20: DRAW 0,6
010: PLOT 1330,20: DRAW 0,6050: PLOT
1335,20: DRAW 0,6070: PLOT 1340,20:
DRAW 0,6090: PLOT 1345,20: DRAW 0,6
110: PLOT 1350,20: DRAW 0,6150: PLOT
1355,20: DRAW 0,6170: PLOT 1360,20:
DRAW 0,6190: PLOT 1365,20: DRAW 0,6
210: PLOT 1370,20: DRAW 0,6250: PLOT
1375,20: DRAW 0,6270: PLOT 1380,20:
DRAW 0,6290: PLOT 1385,20: DRAW 0,6
310: PLOT 1390,20: DRAW 0,6350: PLOT
1395,20: DRAW 0,6370: PLOT 1400,20:
DRAW 0,6390: PLOT 1405,20: DRAW 0,6
410: PLOT 1410,20: DRAW 0,6450: PLOT
1415,20: DRAW 0,6470: PLOT 1420,20:
DRAW 0,6490: PLOT 1425,20: DRAW 0,6
510: PLOT 1430,20: DRAW 0,6550: PLOT
1435,20: DRAW 0,6570: PLOT 1440,20:
DRAW 0,6590: PLOT 1445,20: DRAW 0,6
610: PLOT 1450,20: DRAW 0,6650: PLOT
1455,20: DRAW 0,6670: PLOT 1460,20:
DRAW 0,6690: PLOT 1465,20: DRAW 0,6
710: PLOT 1470,20: DRAW 0,6750: PLOT
1475,20: DRAW 0,6770: PLOT 1480,20:
DRAW 0,6790: PLOT 1485,20: DRAW 0,6
810: PLOT 1490,20: DRAW 0,6850: PLOT
1495,20: DRAW 0,6870: PLOT 1500,20:
DRAW 0,6890: PLOT 1505,20: DRAW 0,6
910: PLOT 1510,20: DRAW 0,6950: PLOT
1515,20: DRAW 0,6970: PLOT 1520,20:
DRAW 0,6990: PLOT 1525,20: DRAW 0,7
010: PLOT 1530,20: DRAW 0,7050: PLOT
1535,20: DRAW 0,7070: PLOT 1540,20:
DRAW 0,7090: PLOT 1545,20: DRAW 0,7
110: PLOT 1550,20: DRAW 0,7150: PLOT
1555,20: DRAW 0,7170: PLOT 1560,20:
DRAW 0,7190: PLOT 1565,20: DRAW 0,7
210: PLOT 1570,20: DRAW 0,7250: PLOT
1575,20: DRAW 0,7270: PLOT 1580,20:
DRAW 0,7290: PLOT 1585,20: DRAW 0,7
310: PLOT 1590,20: DRAW 0,7350: PLOT
1595,20: DRAW 0,7370: PLOT 1600,20:
DRAW 0,7390: PLOT 1605,20: DRAW 0,7
410: PLOT 1610,20: DRAW 0,7450: PLOT
1615,20: DRAW 0,7470: PLOT 1620,20:
DRAW 0,7490: PLOT 1625,20: DRAW 0,7
510: PLOT 1630,20: DRAW 0,7550: PLOT
1635,20: DRAW 0,7570: PLOT 1640,20:
DRAW 0,7590: PLOT 1645,20: DRAW 0,7
610: PLOT 1650,20: DRAW 0,7650: PLOT
1655,20: DRAW 0,7670: PLOT 1660,20:
DRAW 0,7690: PLOT 1665,20: DRAW 0,7
710: PLOT 1670,20: DRAW 0,7750: PLOT
1675,20: DRAW 0,7770: PLOT 1680,20:
DRAW 0,7790: PLOT 1685,20: DRAW 0,7
810: PLOT 1690,20: DRAW 0,7850: PLOT
1695,20: DRAW 0,7870: PLOT 1700,20:
DRAW 0,7890: PLOT 1705,20: DRAW 0,7
910: PLOT 1710,20: DRAW 0,7950: PLOT
1715,20: DRAW 0,7970: PLOT 1720,20:
DRAW 0,7990: PLOT 1725,20: DRAW 0,8
010: PLOT 1730,20: DRAW 0,8050: PLOT
1735,20: DRAW 0,8070: PLOT 1740,20:
DRAW 0,8090: PLOT 1745,20: DRAW 0,8
110: PLOT 1750,20: DRAW 0,8150: PLOT
1755,20: DRAW 0,8170: PLOT 1760,20:
DRAW 0,8190: PLOT 1765,20: DRAW 0,8
210: PLOT 1770,20: DRAW 0,8250: PLOT
1775,20: DRAW 0,8270: PLOT 1780,20:
DRAW 0,8290: PLOT 1785,20: DRAW 0,8
310: PLOT 1790,20: DRAW 0,8350: PLOT
1795,20: DRAW 0,8370: PLOT 1800,20:
DRAW 0,8390: PLOT 1805,20: DRAW 0,8
410: PLOT 1810,20: DRAW 0,8450: PLOT
1815,20: DRAW 0,8470: PLOT 1820,20:
DRAW 0,8490: PLOT 1825,20: DRAW 0,8
510: PLOT 1830,20: DRAW 0,8550: PLOT
1835,20: DRAW 0,8570: PLOT 1840,20:
DRAW 0,8590: PLOT 1845,20: DRAW 0,8
610: PLOT 1850,20: DRAW 0,8650: PLOT
1855,20: DRAW 0,8670: PLOT 1860,20:
DRAW 0,8690: PLOT 1865,20: DRAW 0,8
710: PLOT 1870,20: DRAW 0,8750: PLOT
1875,20: DRAW 0,8770: PLOT 1880,20:
DRAW 0,8790: PLOT 1885,20: DRAW 0,8
810: PLOT 1890,20: DRAW 0,8850: PLOT
1895,20: DRAW 0,8870: PLOT 1900,20:
DRAW 0,8890: PLOT 1905,20: DRAW 0,8
910: PLOT 1910,20: DRAW 0,8950: PLOT
1915,20: DRAW 0,8970: PLOT 1920,20:
DRAW 0,8990: PLOT 1925,20: DRAW 0,9
010: PLOT 1930,20: DRAW 0,9050: PLOT
1935,20: DRAW 0,9070: PLOT 1940,20:
DRAW 0,9090: PLOT 1945,20: DRAW 0,9
110: PLOT 1950,20: DRAW 0,9150: PLOT
1955,20: DRAW 0,9170: PLOT 1960,20:
DRAW 0,9190: PLOT 1965,20: DRAW 0,9
210: PLOT 1970,20: DRAW 0,9250: PLOT
1975,20: DRAW 0,9270: PLOT 1980,20:
DRAW 0,9290: PLOT 1985,20: DRAW 0,9
310: PLOT 1990,20: DRAW 0,9350: PLOT
1995,20: DRAW 0,9370: PLOT 2000,20:
DRAW 0,9390: PLOT 2005,20: DRAW 0,9
410: PLOT 2010,20: DRAW 0,9450: PLOT
2015,20: DRAW 0,9470: PLOT 2020,20:
DRAW 0,9490: PLOT 2025,20: DRAW 0,9
510: PLOT 2030,20: DRAW 0,9550: PLOT
2035,20: DRAW 0,9570: PLOT 2040,20:
DRAW 0,9590: PLOT 2045,20: DRAW 0,9
610: PLOT 2050,20: DRAW 0,9650: PLOT
2055,20: DRAW 0,9670: PLOT 2060,20:
DRAW 0,9690: PLOT 2065,20: DRAW 0,9
710: PLOT 2070,20: DRAW 0,9750: PLOT
2075,20: DRAW 0,9770: PLOT 2080,20:
DRAW 0,9790: PLOT 2085,20: DRAW 0,9
810: PLOT 2090,20: DRAW 0,9850: PLOT
2095,20: DRAW 0,9870: PLOT 2100,20:
DRAW 0,9890: PLOT 2105,20: DRAW 0,9
910: PLOT 2110,20: DRAW 0,9950: PLOT
2115,20: DRAW 0,9970: PLOT 2120,20:
DRAW 0,9990: PLOT 2125,20: DRAW 1,0
010: PLOT 2130,20: DRAW 1,0050: PLOT
2135,20: DRAW 1,0070: PLOT 2140,20:
DRAW 1,0090: PLOT 2145,20: DRAW 1,0
110: PLOT 2150,20: DRAW 1,0150: PLOT
2155,20: DRAW 1,0170: PLOT 2160,20:
DRAW 1,0190: PLOT 2165,20: DRAW 1,0
210: PLOT 2170,20: DRAW 1,0250: PLOT
2175,20: DRAW 1,0270: PLOT 2180,20:
DRAW 1,0290: PLOT 2185,20: DRAW 1,0
310: PLOT 2190,20: DRAW 1,0350: PLOT
2195,20: DRAW 1,0370: PLOT 2200,20:
DRAW 1,0390: PLOT 2205,20: DRAW 1,0
410: PLOT 2210,20: DRAW 1,0450: PLOT
2215,20: DRAW 1,0470: PLOT 2220,20:
DRAW 1,0490: PLOT 2225,20: DRAW 1,0
510: PLOT 2230,20: DRAW 1,0550: PLOT
2235,20: DRAW 1,0570: PLOT 2240,20:
DRAW 1,0590: PLOT 2245,20: DRAW 1,0
610: PLOT 2250,20: DRAW 1,0650: PLOT
2255,20: DRAW 1,0670: PLOT 2260,20:
DRAW 1,0690: PLOT 2265,20: DRAW 1,0
710: PLOT 2270,20: DRAW 1,0750: PLOT
2275,20: DRAW 1,0770: PLOT 2280,20:
DRAW 1,0790: PLOT 2285,20: DRAW 1,0
810: PLOT 2290,20: DRAW 1,0850: PLOT
2295,20: DRAW 1,0870: PLOT 2300,20:
DRAW 1,0890: PLOT 2305,20: DRAW 1,0
910: PLOT 2310,20: DRAW 1,0950: PLOT
2315,20: DRAW 1,0970: PLOT 2320,20:
DRAW 1,0990: PLOT 2325,20: DRAW 1,1
010: PLOT 2330,20: DRAW 1,1050: PLOT
2335,20: DRAW 1,1070: PLOT 2340,20:
DRAW 1,1090: PLOT 2345,20: DRAW 1,1
110: PLOT 2350,20: DRAW 1,1150: PLOT
2355,20: DRAW 1,1170: PLOT 2360,20:
DRAW 1,1190: PLOT 2365,20: DRAW 1,1
210: PLOT 2370,20: DRAW 1,1250: PLOT
2375,20: DRAW 1,1270: PLOT 2380,20:
DRAW 1,1290: PLOT 2385,20: DRAW 1,1
310: PLOT 2390,20: DRAW 1,1350: PLOT
2395,20: DRAW 1,1370: PLOT 2400,20:
DRAW 1,1390: PLOT 2405,20: DRAW 1,1
410: PLOT 2410,20: DRAW 1,1450: PLOT
2415,20: DRAW 1,1470: PLOT 2420,20:
DRAW 1,1490: PLOT 2425,20: DRAW 1,1
510: PLOT 2430,20: DRAW 1,1550: PLOT
2435,20: DRAW 1,1570: PLOT 2440,20:
DRAW 1,1590: PLOT 2445,20: DRAW 1,1
610: PLOT 2450,20: DRAW 1,1650: PLOT
2455,20: DRAW 1,1670: PLOT 2460,20:
DRAW 1,1690: PLOT 2465,20: DRAW 1,1
710: PLOT 2470,20: DRAW 1,1750: PLOT
2475,20: DRAW 1,1770: PLOT 2480,20:
DRAW 1,1790: PLOT 2485,20: DRAW 1,1
810: PLOT 2490,20: DRAW 1,1850: PLOT
2495,20: DRAW 1,1870: PLOT 2500,20:
DRAW 1,1890: PLOT 2505,20: DRAW 1,1
910: PLOT 2510,20: DRAW 1,1950: PLOT
2515,20: DRAW 1,1970: PLOT 2520,20:
DRAW 1,1990: PLOT 2525,20: DRAW 1,2
010: PLOT 2530,20: DRAW 1,2050: PLOT
2535,20: DRAW 1,2070: PLOT 2540,20:
DRAW 1,2090: PLOT 2545,20: DRAW 1,2
110: PLOT 2550,20: DRAW 1,2150: PLOT
2555,20: DRAW 1,2170: PLOT 2560,20:
DRAW 1,2190: PLOT 2565,20: DRAW 1,2
210: PLOT 2570,20: DRAW 1,2250: PLOT
2575,20: DRAW 1,2270: PLOT 2580,20:
DRAW 1,2290: PLOT 2585,20: DRAW 1,2
310: PLOT 2590,20: DRAW 1,2350: PLOT
2595,20: DRAW 1,2370: PLOT 2600,20:
DRAW 1,2390: PLOT 2605,20: DRAW 1,2
410: PLOT 2610,20: DRAW 1,2450: PLOT
2615,20: DRAW 1,2470: PLOT 2620,20:
DRAW 1,2490: PLOT 2625,20: DRAW 1,2
510: PLOT 2630,20: DRAW 1,2550: PLOT
2635,20: DRAW 1,2570: PLOT 2640,20:
DRAW 1,2590: PLOT 2645,20: DRAW 1,2
610: PLOT 2650,20: DRAW 1,2650: PLOT
2655,20: DRAW 1,2670: PLOT 2660,20:
DRAW 1,2690: PLOT 2665,20: DRAW 1,2
710: PLOT 2670,20: DRAW 1,2750: PLOT
2675,20: DRAW 1,2770: PLOT 2680,20:
DRAW 1,2790: PLOT 2685,20: DRAW 1,2
810: PLOT 2690,20: DRAW 1,2850: PLOT
2695,20: DRAW 1,2870: PLOT 2700,20:
DRAW 1,2890: PLOT 2705,20: DRAW 1,2
910: PLOT 2710,20: DRAW 1,2950: PLOT
2715,20: DRAW 1,2970: PLOT 2720,20:
DRAW 1,2990: PLOT 2725,20: DRAW 1,3
010: PLOT 2730,20: DRAW 1,3050: PLOT
2735,20: DRAW 1,3070: PLOT 2740,20:
DRAW 1,3090: PLOT 2745,20: DRAW 1,3
110: PLOT 2750,20: DRAW 1,3150: PLOT
2755,20: DRAW 1,3170: PLOT 2760,20:
DRAW 1,3190: PLOT 2765,20: DRAW 1,3
210: PLOT 2770,20: DRAW 1,3250: PLOT
2775,20: DRAW 1,3270: PLOT 2780,20:
DRAW 1,3290: PLOT 2785,20: DRAW 1,3
310: PLOT 2790,20: DRAW 1,3350: PLOT
2795,20: DRAW 1,3370: PLOT 2800,20:
DRAW 1,3390: PLOT 2805,20: DRAW 1,3
410: PLOT 2810,20: DRAW 1,3450: PLOT
2815,20: DRAW 1,3470: PLOT 2820,20:
DRAW 1,3490: PLOT 2825,20: DRAW 1,3
510: PLOT 2830,20: DRAW 1,3550: PLOT
2835,20: DRAW 1,3570: PLOT 2840,20:
DRAW 1,3590: PLOT 2845,20: DRAW 1,3
610: PLOT 2850,20: DRAW 1,3650: PLOT
2855,20: DRAW 1,3670: PLOT 2860,20:
DRAW 1,3690: PLOT 2865,20: DRAW 1,3
710: PLOT 2870,20: DRAW 1,3750: PLOT
2875,20: DRAW 1,3770: PLOT 2880,20:
DRAW 1,3790: PLOT 2885,20: DRAW 
```

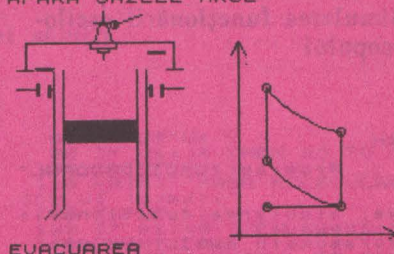


TIMPUL -3- (APRINDEREA SI DETENTA)  
SE PRODUCE O SCINTEIE INTR-  
ELECTROZII BUJIEI CARE APRINDE  
AMESTECUL CARBURANT.GAZELE  
IMPING PISTONUL SPRE P.M. INF.



APRINDEREA SI DETENTA

TIMPUL -4- (EVACUAREA)  
SUPAPA DE EVACUARE SE DESCHIDE.  
PISTONUL SE MISCA DIN P.M. INF.  
SPRE P.M. SUPERIOR SI IMPINGE  
AFARA GAZELE ARSE



EVACUAREA

### 18.33. Program „SIM COM”

Tema: Schema funcțională a unui comutator electronic.

1 REM

Program "SIM COM"

2 REM

1' Tema: Schema funcțională  
la a a unui comutator electronic

3 REM

2' Program BASIC (HC-85)

10 PRINT "Schema funcțională a  
comutator-ului electronic ară-  
ta astfel :"

20 PLOT 50,150: DRAW 40,0: DRA  
W 0,-40: DRAW -40,0: DRAW 0,40

30 PLOT 50,100: DRAW 40,0: DRA  
W 0,-25: DRAW -40,0: DRAW 0,25

40 PLOT 70,120: DRAW 0,-20

50 PLOT 50,55: DRAW 40,0: DRA  
W 0,-40: DRAW -40,0: DRAW 0,40

60 PLOT 50,140: DRAW -35,0: DRA  
W -5,3: DRAW 0,-6: DRAW 5,3

70 PLOT 70,75: DRAW 0,-20

80 PLOT 50,35: DRAW -35,0: DRA  
W -5,3: DRAW 0,-6: DRAW 5,3

90 PRINT AT 3,2: "In 1": AT 16,2

100 PLOT 70,110: DRAW -3,-5: DRA  
W 6,0: DRAW -3,5: PLOT 70,60: D

110 PLOT 90,140: DRAW 0,55: DRA  
W 0,-105: DRAW -55,0: PLOT 145,

90: DRAW 20,0: DRAW 0,3: DRAW 5,

-3: DRAW -5,-3: DRAW 0,3

120 PRINT AT 3,8: "1": AT 16,8: "3

": AT 10,8: "2"

130 PRINT AT 3,20: "1 -Blocul de

": AT 4,20: "Intrare 1": AT 6,20: "2

-Astabil": AT 8,20: "3 -Blocul de

": AT 9,20: "Intrare 2"

140 PAUSE 0: STOP



## 18.34. Program „Osciloscopul” 18.35. Program „Imagine osciloscop

**Tema:** Simularea funcționării osciloscopului

1. REM

2. Program 'OSCILOSCOPUL'

1'. Tema: Simularea funcționării osciloscopului  
2'. Programul în BASIC (HC-85)

```

10 REM Functionare simplificata
20 PRINT AT 1,1;"TUB CATODIC":
PRINT AT 11,13;"AMPLI": PRINT AT
T 12,13;"FICA": PRINT AT 13,13;"
TOR": PRINT AT 18,2;"BAZA": PRIN
T AT 19,3;"DE": PRINT AT 20,2;"T
IMP"
30 PLOT 10,155: DRAW 105,0: DR
AW 65,15: DRAW 0,-60: DRAW -65,1
5: DRAW -105,0: DRAW 0,30
40 PLOT 195,110: DRAW 0,60: DR
AW 59,0: DRAW 0,-60: DRAW -59,0:
50 PLOT 75,130: DRAW 0,-25: DR
AW 90,0: DRAW 0,-25: DRAW -5,0:
PLOT 100,60: DRAW 0,35: DRAW 60,
0: DRAW 0,-35: DRAW -60,0
60 PLOT 100,130: DRAW 0,-20: D
RAW 70,0: DRAW 0,-55: DRAW -95,0
: DRAW 0,-35: PLOT 5,5: DRAW 0,3
5: DRAW 65,0: DRAW 0,-35: DRAW -
65,0: PLOT 70,20: DRAW 5,0
70 LET X1=0
80 FOR N=1 TO 30
90 PLOT 90+N*X1,10+N
100 PLOT 10+N*X1,80+10*SIN (N*P
I/15): PLOT 180+N*X1,80+20*SIN (
N*PI/15)
110 PLOT 10,140: DRAW 65,0: DRA
W 99,20*SIN (N*PI/15): PAUSE 5:
DRAW OVER 1;-99,-1*(20*SIN (N*PI
/15))
120 PLOT 210+N,140+20*SIN (N*PI
/15)
130 IF INKEY$="1" THEN LET N=30
: BEEP .1,10: GO TO 200
140 NEXT N
150 FOR N=1 TO 30 : PLOT OVER 1
: 210+N,140+20*SIN (N*PI/15): NEX
T N
160 PLOT 120,40: DRAW 0,-30: LE
T X1=X1+30
170 IF X1<60 THEN GO TO 80
180 PLOT 150,40: DRAW 0,-30
190 STOP
200

```

**Tema:** Simularea formării imaginii pe ecranul osciloscopului

1 REM

2. Program 'OSCILOSCOPUL'

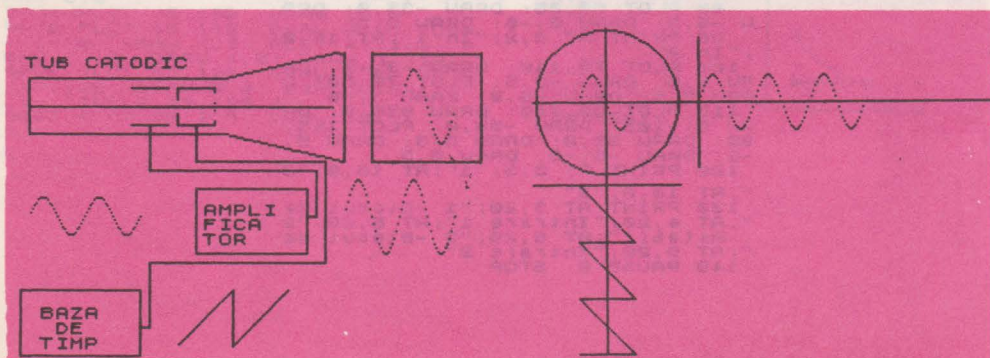
1'. Tema: Simularea pe calculator a formării imaginii pe ecranul osciloscopului.

2'. Programul în BASIC (HC-85)

```

10 REM IMAGINI
20 BORDER 7: INK 0: PAPER 7: C
LS
30 CLS : LET k=0: PRINT AT 0,0
: INK 0: PAPER 7: "FORMAREA I
MAGIILOR PE ECRAN PRIN COMPU
NEREA TENSIUNILOR BAZA DE
TIMP SI CEA STUDIATA ": PRINT
40 PRINT: PRINT "COMPUNEM TEN
SIUNILE ELIMININD TIMPUL DIN R
ELATII"
50 PRINT AT 21,0: FLASH 1:"TAS
TATI ORICE PENTRU A CONTINUA.":
PAUSE 0: CLS
60 PLOT 0,135: DRAW 250,0: PLO
T 40,175: DRAW 0,-175: PLOT 90,1
00: DRAW 0,70: PLOT 0,90: DRAW 8
0,0
70 CIRCLE 40,135,39
80 LET m=90: LET d=0: LET w=1:
LET h=135: LET t=0
90 FOR n=1 TO 30
100 PLOT 90+n+t,h+15*SIN (n*PI/
(15*w)+d)
110 PLOT INK k*2;25+n,h+15*SIN
(n*PI/(15*w)+d)
120 INK 0
130 PLOT 25+n,m-n
140 NEXT n
150 DRAW -30,0
160 LET m=m-30: LET t=t+30: IF
m<20 THEN GO TO 190
170 LET k=k+1
180 GO TO 90
190 PAUSE 100: GO SUB 9990: CLS

```





## 18.36. Program „Normalitatea”

**Tema:** Normalitatea unei soluții

```

0>REM
PROGRAM "NORMALITATEA"

1 REM
1' Tema: Program de cal
culare a normalitatii unei solut
ii
2 REM
2' Program BASIC (HC-85)
10 CLS
20 PRINT "NORMALITATEA SOLUTII
LOR DE ACID"
30 PRINT "MASA SUBSTANTEI DIZO
LUATE, IN GRAME"
40 INPUT md
50 IF md<=0 THEN GO TO 40
60 PRINT "md=";md
70 PRINT "VOLUMUL SOLUTIEI (cm
+3):"

80 INPUT Us
90 IF Us<=0 THEN GO TO 80
100 PRINT "Us=";Us
110 PRINT "MASA MOLECULARA A AC
IDULUI (g/md):"
120 INPUT ma
130 IF ma<=0 THEN GO TO 120
140 PRINT "ma=";ma
150 PRINT "NUMARUL DE PROTONI D
E HIDROGEN CEDATI DE ACID"
160 INPUT na
170 IF na<=0 THEN GO TO 160
180 PRINT "na=";na
190 PRINT "ECHIVALENT-GRAM AL U
NUI ACID:"
200 LET Eg acid=ma/na
210 PRINT "Eg acid=";Eg acid
220 PRINT "NUMARUL DE ECHIVALEN
TI DIN SOLUTIE:"
230 LET ea=md/Eg acid
240 PRINT "ea=";ea
250 PRINT "NORMALITATEA ACIDULU
I:"
260 LET Na=ea/Us
270 PRINT "Na=";Na
280 STOP

```

## 18.37. Program „Molaritatea”

**Tema:** Calcularea molarității unei soluții

```

1>REM
1' Program 'MOLARITATEA'

1' Tema: Program de calculare
a molaritatii unei substante
2 REM
2' Program BASIC (HC-85)
10 CLS
20 PRINT "MASA SUBSTANTEI DIZO
LUATE, IN GRAME"
30 INPUT md
40 IF md<=0 THEN GO TO 30
50 PRINT "md=";md

60 PRINT "MASA MOLECULARA A SU
BSTANTEI, IN GRAME"
70 INPUT M
80 IF M<=0 THEN GO TO 70
90 PRINT "M=";M
100 PRINT "NUMARUL DE MOLII DIN
SOLUTIE:"
110 LET n=md/M: PRINT "n=";n
120 PRINT "VOLUMUL SOLUTIEI IN
LITRI"
130 INPUT Us
140 IF Us<=0 THEN GO TO 130
150 PRINT "Us=";Us
160 PRINT "MOLARITATEA:"
170 LET m=n/Us
180 PRINT "m=";m
190 STOP

```



## Tema: Studiul reactivității elementelor chimice

```

1 REM 28) Program "SERIA REACTIVITA
2 REM TII ELEMENTELOR CHIMICE"
3 REM 1'. Tema : Studiul reactivita
4 REM tii chimice a metalelor
5 REM 2'. Program BASIC (HC-85)
10 CLS
20 GO SUB 370
30 LET a=95: LET b=150: GO SUB 220
40 LET a=55: LET b=150: GO SUB 220
50 LET a=15: LET b=150: GO SUB 220
60 LET a=222: LET b=150: GO SUB 220
70 LET a=182: LET b=150: GO SUB 220
80 LET a=142: LET b=150: GO SUB 220
90 PRINT AT 9,14;"=>"
100 PRINT AT 20,1;"Zn"
110 PRINT AT 18,2; OVER 1;"FeSO4"
120 PRINT AT 20,17;"Fe";AT 18,18;"ZnSO4"
130 PAUSE 40: PRINT AT 20,6;"Zn"
140 PRINT AT 18,7; OVER 1;"CuSO4"
150 PRINT AT 20,22;"Cu";AT 18,23;"ZnSO4"
160 PAUSE 40: PRINT AT 20,11;"Zn";
170 PRINT AT 18,12; OVER 1;"HgCl2"
180 PRINT AT 20,27;"Hg";AT 18,28;"ZnCl"
190 PRINT #1; FLASH 1;"Tastati orice."
200 PAUSE 0: CLS : GO TO 620
210 STOP
220 PLOT a+10,b: DRAW 0,-100
230 PLOT a+10,b-85: DRAW -20,0
240 FOR i=PI/2 TO 3*PI/2 STEP .1
250 PLOT a+10*SIN i,b-100+10*COS i
260 NEXT i
270 DRAW 0,100: PLOT a+7,b-89
280 DRAW -5,0: PLOT a+4,b-93: DRAW -5,0
290 PLOT a+6,b-99: DRAW -7,0
300 PLOT a+8,b-93: DRAW 7,0: DRAW 0,-25
310 DRAW 2,2: DRAW -4,0: DRAW 2,-2
320 FOR i=0 TO 4
330 CIRCLE a,b-105,i: NEXT i
340 PLOT a,b-105: DRAW 0,-25
350 DRAW 2,2: DRAW -4,0: DRAW 2,-2
360 RETURN
370 CLS
380 PRINT " Experimental s-a stabilit";
390 PRINT " ca reactivitatea metalelor";
400 PRINT " este in ordinea descreasca";
410 PRINT "toare, a caracterului lor";
420 PRINT " electropozitiv."
430 PRINT AT 4,3;"Vom face experientele:
440 PRINT " a) Introducem cite o ";
450 PRINT "granula de Zn in solutiile ";
460 PRINT "de :FeSO4,CuSO4,HgCl2.Dupa ";
470 PRINT "aproximativ 2 minute vom ob";
480 PRINT "serva ca Zn a substituit ";
490 PRINT "Fe,Cu,Hg din sarurile lor,";
500 PRINT "acope-rindu-se cu un strat ";
510 PRINT "cenusiu de Fe,roschat de Cu,";
520 PRINT "lucios de Hg. b) Introduc
530 PRINT "cem intr-o eprubeta Zn si in ";
540 PRINT "alta Pb cu solutiile de Pb";
550 PRINT "(NO3)2 respectiv Zn(NO3)2 si";
560 PRINT " vom constata ca Zn a subs";
570 PRINT "tituit Pb din Pb(NO)3 , in ";
580 PRINT "schimb Pb nua substituit."
590 PRINT #1; FLASH 1;"Tastati orice."
600 PAUSE 0: CLS
610 GO TO 30
620 PRINT "Ecuatiile reactiilor chimi";
630 PRINT "ce sint:"
640 PRINT AT 7,0;"1) Zn + FeSO4 -> ";
650 PRINT "Fe + ZnSO4"
660 PRINT AT 12,2;" Pb + CuSO4 -> ";
670 PRINT "Cu + ZnSO4"
680 PRINT AT 17,2;" Zn + HgCl2 ->";
690 PRINT " Hg + ZnCl2"
700 PRINT AT 9,2;"zinc";AT 9,7;"sulfat"
710 PRINT AT 10,7;"de fier"
720 PRINT AT 9,16;"fier";AT 9,22;"sulfat"
730 PRINT AT 10,22;"de zinc"
740 PRINT AT 14,2;"zinc";AT 14,7;"sulfat"
750 PRINT AT 15,6;"de cupru"
760 PRINT AT 14,15;"cupru"
770 PRINT AT 14,22;"sulfat"
780 PRINT AT 15,22;"de zinc"
790 PRINT AT 19,2;"zinc"
800 PRINT AT 19,7;"clorura"
810 PRINT AT 20,6;"de mercur"
820 PRINT AT 19,15;"mercur"
830 PRINT AT 19,22;"clorura"
840 PRINT AT 20,22;"de zinc"
850 PRINT #1; FLASH 1;"Tastati orice."
860 PAUSE 0
870 CLS
880 CLS : LET a=15: LET b=150
890 GO SUB 220: LET a=75: LET b=150
900 GO SUB 220: LET a=160: LET b=150
910 GO SUB 220: LET a=220: LET b=150
920 GO SUB 220
930 PRINT AT 10,15;"=>"
940 PRINT AT 20,1;"Zn"
950 PRINT AT 18,2; OVER 1;"Pb(NO3)"
960 PRINT AT 20,19;"Pb"
970 PRINT AT 18,20; OVER 1;"Zn(NO3)2"
980 PAUSE 45
990 PRINT AT 20,10;"Pb"
1000 PRINT AT 18,11; OVER 1;"Zn(NO3)2"
1010 PRINT AT 20,27;"Pb"
1020 PRINT AT 21,23;"Zn(NO3)2";
1030 PRINT #1; FLASH 1;"Tastati orice."
1040 PAUSE 0: CLS
1050 PRINT " Pentru cea de-a doua expe";
1060 PRINT "rientaavem reactiile:"
1070 PRINT AT 5,1;"Zn + Pb(NO3)2 -> ";
1080 PRINT "Zn(NO3)2 + Pb"
1090 PRINT AT 14,2;"Pb + Zn(NO3)2 ->"
1100 PRINT AT 7,0;"zinc";AT 7,6;"azotat";
1110 PRINT " de ";
1120 PRINT " plumb";AT 7,17;"azotat de";
1130 PRINT " zinc"
1140 PRINT AT 7,27;"plumb"
1150 PRINT AT 16,0;" Aceasta reactie "
1160 PRINT "nu are loc"
1170 PRINT #1; FLASH 1;"Tastati orice."
1180 PAUSE 0: CLS
1190 PRINT " Experimental s-a stabilit ";

```



```

1200 PRINT "reactivitatea elementelor ";
1210 PRINT "este in ordinea descresca";
1220 PRINT "toare a caracterului lor ";
1230 PRINT "electropozitiv."
1240 PRINT AT 18,30;"Au";AT 18,27;"Pt"
1250 PRINT AT 18,24;"Ag";AT 17,22;"Hg"
1260 PRINT AT 16,20;"Cu"
1270 PRINT AT 15,18; FLASH 1;"H2"
1280 PRINT AT 14,16;"Pb";AT 13,14;"Fe"
1290 PRINT AT 12,12;"Zn";AT 11,10;"Al"
1300 PRINT AT 10,8;"Mg";AT 9,6;"Na"
1310 PRINT AT 8,4;"Ca";AT 7,4;"K"
1320 PLOT 254,16: DRAW -240,100,-.9
1330 DRAW -10,0: DRAW 35,20
1340 PLOT 255,16: DRAW 0,22
1350 DRAW -190,78,-.6: DRAW 10,0
1360 DRAW -35,20
1370 STOP

```

### 18.39. Program „Ultima cifră”

#### Funcția „ultima cifră”

În lucrare se introduce funcția:

$$\varepsilon : \mathbb{N} \times \mathbb{N} - (0,0) \rightarrow A, \text{ unde } A = \{k \in \mathbb{N} \mid 0 \leq k < 10\}$$

care asociază perechii  $(a,n) \in \mathbb{N} \times \mathbb{N} - (0,0)$  cifra  $k \in A$ , ce reprezintă ultima cifră a numărului  $a^n$ .

De asemenea, s-a studiat o extindere a funcției pe  $\mathbb{Z}$  astfel:

$$\varepsilon : \mathbb{Z} \times \mathbb{N} - (0,0) \rightarrow A$$

și o legătură între funcția „ultima cifră” și relația de congruență (mod  $m$ ). S-a observat o anumită periodicitate în calculul ultimei cifre a unei puteri, periodicitate exprimată de următorul rezultat:

Pentru orice pereche  $(a, n) \in \mathbb{N} \times \mathbb{N} - (0,0)$  avem:

$$1. (\exists) T \in A^* \mid \varepsilon(a, n) = \varepsilon(a, n+T) = \varepsilon(a, n+kT), n+kT=0 \text{ și } k \in \mathbb{N}.$$

Numim  $T$  perioada atașată funcției  $\varepsilon(a,n)$  când  $n$  rămâne constant.

$$2. \varepsilon(a, n) = \varepsilon(a + m \cdot 10^k, n) \text{ unde } m, k \in \mathbb{N}^*$$

Numim  $\tau = m \cdot 10^k$ , perioada atașată funcției  $\varepsilon(a,n)$  când  $n$  rămâne constant. Notăm cu  $a_k = \varepsilon(a, n)$ . În acest caz avem următorul tabel, care ne dă perioada  $T$ :

---

\* Lucrare prezentată de elevul Bălan Radu, clasa a-X-a, Liceul „Dimitrie Cantemir” — București, la faza națională a Sesiunii de comunicări științifice și referate a elevilor, mai, 1985, Iași.



a    0   1   2   3   4   5   6   7   8   9

T    1   1   4   4   2   1   1   4   4   2

Se observă o anumită periodicitate Primele cinci numere se repetă.  
Pentru aplicații avem următoarele reguli de calcul:

$$3. \quad \varepsilon(a \cdot b, n) = \varepsilon(\varepsilon(a, n) \cdot \varepsilon(b, n), 1)$$

$$4. \quad \varepsilon(a^n + b^n, 1) = \varepsilon(\varepsilon(a, n) + \varepsilon(b, n), 1)$$

5.  $\varepsilon(a + b, n) = \varepsilon(\varepsilon(a, 1) + \varepsilon(b, n), n)$  și de asemenea, generalizarea acestora pentru m numere.

Se observa, de asemenea, că:

$$n = 10. \quad [n/10] + \varepsilon(n, 1),$$

rezultat care sugerează extinderea funcției „ultima cifră” pe  $\mathbb{Z}$  prin următoarea funcție:

$E : \mathbb{Z} \times \mathbb{N} \rightarrow (0, 10) \rightarrow A$ , astfel:

$$E(a, n) = \begin{cases} \varepsilon(a, n), & a \geq 0 \\ \varepsilon(|a|, n), & a < 0 \text{ și } n = 2k, k \in \mathbb{N} \\ \varepsilon(10 - \varepsilon(|a|, n), 1), & a < 0 \text{ și } n = 2k + 1 \end{cases}$$

În acest caz avem deci:

$$n = 10. \quad [n/10] + E(n, 1), \quad (\forall) n \in \mathbb{Z}$$

de asemenea, regulile stabilite pentru funcția „ $\varepsilon$ ” rămân adevărate și pentru funcția  $E$ , în plus:

$$6. \quad E(a^n - b^n, 1) = E(E(a, n) - E(b, n), 1)$$

$$7. \quad E(a - b, n) = \varepsilon(10 + E(a, 1) - E(b, 1), n).$$

Legătura între funcția „ultima cifră” și relația de congruență (mod m) este dată de:

$$\varepsilon(n, 1) = n \pmod{10}$$

În plus, atunci când trecem un număr din baza 10 într-o altă bază (spre exemplu k) avem:

$$\varepsilon(a_{(k)}, 1) = a \pmod{k}$$

unde  $\varepsilon(a_{(k)}, 1)$  reprezintă ultima cifră a lui  $a_{(k)}$  (a trecut în baza k), scrisă apoi în baza 10.

de asemenea, când lucrăm într-o altă bază avem:

$$a_{(k)} = \varepsilon_{(k)}(a_{(k)}, 1) + k \cdot [a_{(k)}/k]_{(k)}$$

Aplicând teorema lui Euler pentru numerele naturale:  $(a, k) = 1$  avem:



$$a^{(k)} \equiv 1 \pmod{k}$$

Deci:

$$\epsilon^{(k)}(a_{(k)}, 1) \equiv 1 \pmod{k}.$$

În baza 10, pentru numerele  $(a, 10) = 1$ , deci:  $\epsilon(a, 1) \in \{1, 3, 7, 9\}$  avem:

$$\epsilon^{(10)}(a, 1) \equiv 1 \pmod{10}.$$

Deci:

$$\epsilon^4(a, 1) \equiv 1 \pmod{10}.$$

Sau:  $a^4 \equiv 1 \pmod{10}$

ceea ce arată că perioada maximă a numerelor a căror ultimă cifră este 1, 3, 7 sau 9 este 4.

```

10 CLS
11 PRINT "PROGRAM DE CALCULARE
A ULTIMEI CIFRE A UNEI P
UTERI"
12 PRINT : PRINT "INTRODUCETI
BAZA A="
13 INPUT LINE A$
14 IF SGN VAL A$ < 0 THEN GO TO
260
40 LET N=LEN A$
50 LET B$=A$(N)
54 LET A2=VAL B$
55 CLS : PRINT "PROGRAM DE CAL
CULARE A ULTIMEI CIFRE A
UNEI PUTERI" : PRINT
56 PRINT : PRINT "INTRODUCETI
EXPONENTIAL E="
57 INPUT LINE E$
59 CLS : PRINT "PROGRAM DE CAL
CULARE A ULTIMEI CIFRE A
UNEI PUTERI" : PRINT "BAZ
A ESTE A=" : PRINT "EX
PONENTIALUL ESTE E=" : PRINT
74 LET M=LEN E$
76 IF M>1 THEN GO TO 80
77 IF M=1 THEN LET E3=VAL E$(
M) : LET E2=0 : GO TO 98
80 LET M=M-1
82 LET G$=E$(M+1)
87 LET E3=VAL G$
90 LET F$=E$(M)
96 LET E2=VAL F$
98 LET E2=10*E2+E3
100 IF VAL A$=0 AND VAL E$=0 TH

```

```

EN PRINT : PRINT "NUMARUL N=0+0
NU ESTE DEFINIT PE MULTIMEA NUME
RELOR REALE" : GO TO 260
120 IF VAL A$=0 THEN PRINT : PR
INT "ULTIMA CIFRA ESTE 0=0" : GO
TO 260
130 IF E2=0 THEN GO TO 150
140 IF E2=0 THEN GO TO 205
150 IF VAL E$=0 THEN PRINT : PR
INT "ULTIMA CIFRA ESTE 1=1" : GO
TO 260
160 LET A1=A2+4
170 LET A2=A1-10*(INT (A1/10))
180 LET A2=INT (A2)
190 PRINT "ULTIMA CIFRA ESTE 0=
";A2
200 GO TO 260
205 LET E2=E2-4*INT (E2/4)+4
207 LET E2=INT (E2)
210 LET P1=A2+E2
220 LET P2=P1-10*INT (P1/10)
230 LET P2=INT (P2+0.1)
240 PRINT : PRINT : PRINT "ULTI
MA CIFRA ESTE 0=" : PRINT P2
250 PRINT : PRINT : PRINT "PRI
NT : PRINT : PRINT "DORITI UN NO
U RULAU ?" : PRINT "DACA DA TASTA
TI /DA/"
270 INPUT A$
280 IF A$="DA" THEN GO TO 1
285 FOR I=1 TO 5
290 CLS : PRINT AT I,10-I;"LA R
EVEDERE" : PRINT AT 10-I,1;"LA REV
EDERE" : BEEP 0.25,I : NEXT I
300 STOP

```

## 18.40. Program „Modulul”

```

5 BORDER 0: PAPER 0: INK 7: C
LS
10 PRINT " FUNCTII MODUL"
20 PRINT : PRINT "1. Găfice d
e functii si relatii"
30 PLOT 0,145: DRAW 255,0: PLO
T 129,10: DRAW 0,143
40 PRINT AT 4,2;"Y=IXI": PRINT
AT 4,10;"Y=-IXI"
50 FOR I=127 TO 1 STEP -2: PLO
T 64,I: NEXT I
60 PLOT 2,72: DRAW 60,0: PLOT
32,50: DRAW 0,73
70 PLOT 66,72: DRAW 60,0: PLOT
96,12: DRAW 0,73
80 FOR I=1 TO 60: PLOT I+2,ABS
(I-30)+72: PLOT I+66,72-ABS (I-
30): NEXT I
90 PRINT AT 8,0;"Y=-X": PRINT
AT 8,5;"Y=X": PRINT AT 16,8;"Y=X
": PRINT AT 16,12;"Y=-X"
95 PRINT AT 7,4;"Y": PRINT AT

```

```

12,6;"X": PRINT AT 11,12;"Y": PR
INT AT 12,15;"X": PAUSE 100
100 PRINT AT 4,19;"YI=X": PRIN
T AT 4,25;"YI=IXI"
110 FOR I=127 TO 2 STEP -2: PLO
T 196,I: NEXT I
120 PLOT 130,72: DRAW 60,0: PLO
T 132,12: DRAW 0,120
130 PLOT 194,72: DRAW 60,0: PLO
T 196,12: DRAW 0,120
132 PRINT AT 6,17;"Y": PRINT AT
12,23;"X": PRINT AT 6,27;"Y": P
RINT AT 11,31;"X"
135 PRINT AT 7,17;"Y=X": PRINT
AT 16,17;"Y=-X": PRINT AT 7,25;"
Y=-X": PRINT AT 16,25;"Y=X"
140 FOR I=0 TO 59: PLOT 132+I,7
2+I: PLOT 132+I,72-I: PLOT 196+I
,102-I: PLOT 196+I,42+I: NEXT I
150 PRINT AT 21,0: FLASH 1 : PE
NTRU A CONTINUA TASTATI ORICE"
160 INPUT INKEY$

```



```

170 IF INKEY$="" THEN GO TO 160
180 CLS
190 PRINT "2. GRAFICUL FUNCTIEI
      f(x)=11x+11-11"
200 PRINT "2.1 MAI INTII, TRASAR
    EA DIRECTA"
210 PLOT 5,30: DRAW 245,0: PLOT
140,20: DRAW 0,120
220 FOR I=5 TO 250
230 IF I<=100 THEN LET Y=130-I
240 IF I<=120 AND I>100 THEN LE
T Y=I-70
250 IF I<=140 AND I>120 THEN LE
T Y=170-I
260 IF I>140 THEN LET Y=I-110
265 IF I=120 THEN PRINT AT 14,1
270 PLOT I,Y
280 NEXT I
290 PRINT AT 19,11;"-2 0"
295 PRINT AT 4,18;"y": PRINT AT
17,30;"x"
300 PRINT AT 20,4: FLASH 1;"PEN
    TRU A CONTINUA"
310 PRINT AT 21,8: FLASH 1;"APA
    SATI ORICE TASTA"
320 INPUT INKEY$
330 IF INKEY$="" THEN GO TO 320
340 CLS
350 PRINT "2.2 FOLOSIREA PROCED
    EULUI TRANSLATIEI PENTRU TRASARE
    A PE ETAPE AGRAFICULUI"
360 PRINT "1) SE REALIZEAZA GRA
    FICUL FUNCTIEI f(x)=11x"
370 PLOT 5,30: DRAW 245,0: PLOT
140,20: DRAW 0,120
380 FOR I=45 TO 235 STEP 2
390 LET Y=30+ABS (I-140)
400 PLOT I,Y
410 NEXT I
415 PRINT AT 19,14;"-1"
417 PRINT AT 4,18;"y": PRINT AT
17,30;"x"
419 PAUSE 500
420 PRINT AT 5,0;"ii) SE TRANSL
    ATERAZA ACEST GRAFICPE AXA OX CU
    -1"
430 FOR I=50 TO 232
440 LET Y=30+ABS (I-140)
450 PLOT I-20,Y
460 NEXT I
470 PAUSE 450: CLS
480 PLOT 5,30: DRAW 245,0: PLOT
140,20: DRAW 0,120
485 PRINT AT 4,18;"y": PRINT AT
17,30;"x"
490 FOR I=30 TO 250 STEP 2
500 LET Y=30+ABS (I-140)
510 PLOT I-20,Y
520 NEXT I
530 PRINT AT 2,0;"iii) SE TRANSL
    ATERAZA ACUM GRAFI- CUL PE AXA (O
    y) CU -1"
540 FOR I=30 TO 250
550 LET Y=30+ABS (I-140)
560 PLOT I-20,Y-20
565 IF I=140 THEN PRINT AT 21,1
1;"(-1,-1)"
570 NEXT I: PAUSE 250: CLS
571 PLOT 5,30: DRAW 245,0: PLOT
140,20: DRAW 0,120
572 FOR I=25 TO 270 STEP 2
574 LET Y=30+ABS (I-140)
576 PLOT I-20,Y-20
577 IF I=141 THEN PRINT AT 21,1
1;"(-1,-1)"
578 NEXT I
579 PRINT AT 5,18;"y": PRINT AT
17,30;"x"
580 PRINT AT 0,0:
iv) PENTRU TRASAREA GRAFICULUI
    FUNCTIEI CERUTE SE <<RIDICA>>,
    PRIN SIMETRIA IN RAPORT CU AXA
    OX, PORTIUNEA NEGATIVA A GRAFI-
    CULUI FUNCTIEI DATE."
585 PRINT AT 5,18;"y": PRINT AT
17,30;"x"
590 FOR I=5 TO 250

```

```

840 NEXT X
850 PRINT AT 0,0;"3.2 Urmeaza t
    rasarea graficului functiei y=1/
    (x+1), pozitivind - prin simetria
    in raport cu axa (OX)-ramura n
    egativa a lui f(x)"
855 PRINT AT 19,26;"G-0"
860 FOR x=1 TO 245
870 IF x<40 THEN LET y=90-x
880 IF x>=40 AND x<80 THEN LET
    y=x+10
890 IF x>=80 AND x<100 THEN LET
    y=170-x
900 IF x>=100 AND x<180 THEN LE
T y=0.25*x+45
910 IF x>=180 AND x<221 THEN LE
T y=-x+270
920 IF x>=221 THEN LET y=x-171
925 IF x=80 THEN PRINT AT 9,7;"
    (-2,2)"
927 IF x=100 THEN PRINT AT 13,1
6;"1"
929 IF x=181 THEN PRINT AT 9,21
;"(3,2)"
930 PLOT x+4,y+3
940 NEXT x
943 PRINT AT 12,26;"G-1"
945 PAUSE 0
946 LET J=3: GO SUB 5000
947 PRINT AT 1,0;"Realizam graf
    icul relatiei y1=1/
    (x) notat R"
948 GO SUB 7000
949 GO SUB 8000
950 CLS: PRINT "4. GRAFICUL RE
    LATIEI "

```



```

960 PRINT "12y-11+12y+11+450R 3
/31X1-4=0": PRINT
970 PRINT "4.1 EXPLICITIND, PE I
NTERVALE, SE OBTINE:"
975 PRINT AT 5,1:

```

```

980 PRINT AT 5,2: "(SQR 3)X+3y-3
=0, X2(0, SQR 3/2): SI YE(1/2, 1)
990 PRINT AT 7,2: "2X-SQR 3=0, X
=SQR 3/2: SI YE(-1/2, 1)
1000 PRINT AT 9,2: "SQR 3X+3y+3=0
X2(0, SQR 3/2): SI YE(-1/2, 1)
1010 PRINT AT 11,2: "SQR 3X-3y+3=
0, X2(-SQR 3/2, 0): SI YE(1/2, 1)

```

```

1020 PRINT AT 13,2: "2X+SQR 3=0,
X=-SQR 3/2: SI YE(-1/2, 1/2)
1030 PRINT AT 15,2: "SQR 3X+3y+3=
0, X2(-SQR 3/2, 0): SI YE(-1/2, 1)
1032 PRINT AT 5,1: "I": PRINT AT
5,1: "I": PRINT AT 6,1: "I": PRINT
AT 7,1: "I": PRINT AT 8,1: "I": P
RINT AT 9,1: "I": PRINT AT 10,1: "
I": PRINT AT 11,1: "I"
1033 PRINT AT 12,1: "I": PRINT AT
13,1: "I": PRINT AT 14,1: "I"
1035 PRINT "NOTAM CU R(X,Y)=12y+
11+12y-11+4 SQR 3/31X1 AVEAM R(X,
Y)=R(X,Y) SI R(-X,Y)=R(X,Y)"
1040 PRINT "-APASATI O TASTA":
PAUSE 0

```

```

1050 CLS: PLOT 5,85: DRAW 245,0
: PLOT 125,5: DRAW 0,165: PRINT
AT 2,14: "y": PRINT AT 10,30: "x":
PRINT AT 10,16: "0"
1060 FOR I=1 TO 50
1064 IF I<25 THEN PLOT 167,85+I
1066 IF I>25 THEN PLOT 167-42/25
*(I-25), I+85
1067 IF I=25 THEN PRINT AT 7,21:
"A(0,6,0,5)"
1068 NEXT I
1069 PRINT AT 4,14: "B(0,1)"
1070 PRINT AT 0,0: "A) FOLOSIM REL
ATIA R(X,Y)=R(X,Y)"
1075 PLOT 5,5: PAUSE 0
1080 FOR I=1 TO 50
1084 IF I<25 THEN PLOT 125-42/2
5*I, 135-I

```

```

1086 IF I>25 THEN PLOT 83,135-I
1087 IF I=25 THEN PRINT AT 7,0: "
C(0,8,0,5)"
1088 NEXT I
1090 PRINT AT 1,0: INVERSE 1: "i
SI ACUM R(X,-Y)=R(X,Y)"
1095 PLOT 5,5: PAUSE 0
1100 FOR I=1 TO 100
1102 IF I<25 THEN PLOT 83,85-I
1103 IF I=25 THEN PRINT AT 15,0:
"D(0,8,-0,5)"
1104 IF I>25 AND I<50 THEN PLOT
83+42/25*(I-25), 85-I
1105 IF I=50 THEN PRINT AT 18,14:
"E(0,1)"
1106 IF I>50 AND I<75 THEN PLOT
125+42/25*(I-50), I-15
1107 IF I=75 THEN PRINT AT 15,21:
"F(0,8,-0,5)"
1108 IF I>75 THEN PLOT 167, I-15
1109 NEXT I
1110 PRINT AT 21,6: "-APASATI PE
O TASTA-"
1111 PAUSE 0
1115 CLS
1120 PRINT "5. GRAFICUL RELATIEI
: (11X1+11Y1-31-31)=1

```

```

X,Y=0"
1130 PRINT: PRINT "R
EALIZAREA GRAFICULUI CONSTITUIE
UN EXERCITIU LABORIOS - IN CAZUL
FOLOSESIRII PROCEDEULUI EXPLICIT
ARIU, PE ACEASTA CALE, VOR REZULT
A 16 ECURATII CARORA LE CORESP
UND 16 SEGMENTE DE DREAPTA."
1140 PRINT: PRINT "P
ROCEDEUL SIMETRIEI ESTE INSA
MULT MAI EFICIENT: I-X1=IXI SI I-
Y1=IYI. ASTFEL, SE REALIZEAZA MAI
INTII GRAFICUL PENTRU CADRANUL I
SI SE FOLOSESC APOI SIMETRIILE
IN RAPORT CU AXELE"
1150 PAUSE 0: CLS: PLOT 5,85: D
RAW 245,0: PLOT 125,5: DRAW 0,16

```

```

5: PRINT AT 2,14: "y": PRINT AT 1
0,30: "x": PRINT AT 10,16: "0"
1160 PLOT 135,85: DRAW 30,30: DR
AW -40,40

```

```

1170 PLOT 125,100: DRAW 15,15: D
RAW -15,15
1180 PRINT AT 7,21: "(4,3)": PRIN
T AT 2,16: "(0,7)": PRINT AT 0,0:
"SIMETRIZAREA IN RAPORT CU (0y)":
: PAUSE 0: PLOT 125,155: DRAW 1
0,-10
1190 PLOT 125,155: DRAW -40,-40:
DRAW 30,-30
1200 PLOT 125,130: DRAW -15,-15:
DRAW 15,-15
1210 PRINT AT 7,4: "(-4,3)":
1220 PRINT AT 0,0: "SIMETRIZAREA
IN RAPORT CU (0x)": : PAUSE 0
1230 PLOT 115,85: DRAW -30,-30:
DRAW 40,-40: DRAW 40,40: DRAW -3
0,30
1240 PRINT AT 20,16: "(0,-7)": AT
15,3: "(-4,-3)": AT 15,21: "(4,-3)"
1340 PLOT 125,70: DRAW 15,15:
DRAW 15,-15: DRAW 15,15: DRAW -1
5,15

```

```

1350 FOR I=1 TO 30 STEP 2
1360 PLOT 141+I, 114-I
1370 NEXT I
1380 PRINT AT 12,20: "5 7"
1390 FOR I=1 TO 30 STEP 2
1400 PLOT 165+I, 115-I
1410 NEXT I
1420 STOP
2000 PRINT "REALIZAM GRAFICUL LU
I Y=f(IXI)"
5000 FOR I=1 TO 5010
5010 LET K=-1
5020 LET K=K+1
5030 IF K=3 THEN GO TO 5240
5040 GO SUB 6000
5140 PRINT AT 1,0: "SE DESENEAZA
ACUM GRAFICUL FUNC TIEI y=f(IXI)
, G-2"

```

```

5145 IF K=1 THEN PRINT AT 3,0: "S
I-DESENEAZA y=-f(IXI), G-3": P
RINT AT 11,24: "G-3"
5147 IF K=2 THEN PRINT AT 1,0: "
SE DESENEAZA ACUM GRAFICELE FUNC
TIIOR DATE DE y=f(IXI), G-4
SI y=-f(IXI), G-5": PRINT
AT 10,16: "G-4": PRINT AT 20,16: "
G-5"
5150 FOR X=1 TO 245
5160 IF X<65 THEN LET Y=-X+75
5170 IF X>65 AND X<120 THEN LET
Y=0,25*X+55
5180 IF X>120 AND X<179 THEN LE
T Y=-0,25*X+55
5190 IF X>179 THEN LET Y=X-170
5200 IF K=0 OR K=1 THEN PLOT X+5
,Y+2
5204 IF K=1 THEN PLOT X+5,97-Y
5206 IF K=2 THEN INK 3: PLOT X+4
,Y+3+ABS(Y-50): PLOT X+4,47-ABS
(Y-50): INK 7

```

```

5210 NEXT X
5220 PRINT AT 12,4: "G-0": IF K=0
OR K=1 THEN PRINT AT 18,3: "G-2"
5235 PAUSE 0: CLS: GO TO 5020
5247 LET K=0
5250 GO SUB 6000
5255 PRINT AT 12,4: "
5260 IF K=0 OR K=1 THEN PRINT AT
1,0: "SE DESENEAZA ACUM GRAFICUL
FUNC TIEI y=f(-IXI), G-6
SI-DE ASEMENEA y=-f(-IXI),
G-7"
5265 IF K=0 THEN PRINT AT 3,0: "

```

```

5267 IF K=2 THEN PRINT AT 1,0: "
SE DESENEAZA ACUM GRAFICELE FUNC
TIIOR DATE DE y=f(-IXI), G-6
y=-f(-IXI), G-9"

```

```

5270 FOR X=1 TO 245
5280 IF X<80 THEN LET Y=90-X
5290 IF X>80 AND X<100 THEN LET
Y=X-70
5300 IF X>100 AND X<120 THEN LE
T Y=-0,25*X+55
5310 IF X>120 AND X<140 THEN LE
T Y=(0,25*X)-5
5320 IF X>140 AND X<160 THEN LE
T Y=-X+170
5330 IF X>160 THEN LET Y=X-150
5332 IF K=1 THEN PLOT X+5,97-Y
5333 IF K=2 THEN INK 3: PLOT X+1
,Y+3+ABS(Y-50): PLOT X+1,47-ABS
(Y-50): INK 7
5335 IF K=0 OR K=1 THEN PLOT X+5
,Y+2

```



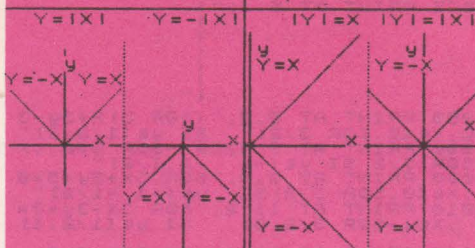
```

5340 NEXT X
5345 PRINT AT 16,25;"4": PRINT A
T 12,4;"": PRINT AT 19,25;"G-
0"
5346 IF K=0 OR K=1 THEN PRINT AT
11,26;"G-6"
5347 IF K=1 THEN PRINT AT 10,20;"
G-7"
5348 IF K=2 THEN PRINT AT 10,3;"
G-8"
5349 PRINT AT 20,3;"G-9"
5350 PAUSE 0
5350 LET K=K+1
5350 IF K=3 THEN GO TO 5390
5350 CLS: GO TO 5250
5350 RETURN
5350 CLS: PRINT "3.";"J:" FIE GR
AFICUL LUI  $y=f(x)$ 
5355 LET J=J+1
5360 PLOT 5,30: DRAW 245,0: PLOT
125,5: DRAW 0,165: PRINT AT 4,1
6;"J" AT 15,31;"X"
5362 PRINT AT 16,16;"0": PRINT A
T 16,8;"-3": PRINT AT 16,2;"-5":
PRINT AT 16,0;"-6"
5363 FOR X=1 TO 245 STEP 2
5364 IF X<80 THEN LET Y=90-X
5365 IF X>=80 AND X<100 THEN LET
Y=X-70
5366 IF X>=100 AND X<180 THEN LE
T Y=-0.25*X+55
5367 IF X>=180 THEN LET Y=X-170
5370 IF X=81 THEN PRINT AT 21,7;"
(-2,2)"
5380 IF X=41 THEN PRINT AT 16,4;"
-4"
5390 IF X=181 THEN PRINT AT 21,2
0;"(3,-2)"
5400 IF X=101 THEN PRINT AT 16,1
2;"-1": PRINT AT 17,16;"-1"
5410 PLOT X+5,Y
5420 IF X=221 THEN PRINT AT 16,2
8;"S"
5430 NEXT X
5435 PRINT AT 12,4;"G-0"
5440 RETURN
5445 CLS: GO SUB 6000: FOR X=1
TO 245
5450 PRINT AT 1,0;"Realizam graf
icul relatiei  $|y|=f(x)$ "
5455 IF X=1 THEN PRINT AT 1,0;"R"
5460 FOR X=1 TO 245
5465 IF X<80 THEN LET Y=90-X
5470 IF X>=80 AND X<100 THEN LET
Y=X-70
5475 IF X>=100 AND X<180 THEN LE
T Y=-0.25*X+55
5480 IF X>=180 THEN LET Y=X-170
5485 PLOT X+5,52+ABS(Y-50): PLO
T X+5,48-ABS(Y-50)
5490 NEXT X
5495 PRINT AT 9,10;"R"
5500 PAUSE 0
5505 RETURN
5510 GO SUB 6000
5515 PRINT AT 1,0;"Realizam graf
icul relatiei  $|y|=f(x)$ "
5520 IF X=1 THEN PRINT AT 1,0;"R"
5525 FOR X=1 TO 245
5530 IF X<80 THEN LET Y=90-X
5535 IF X>=80 AND X<100 THEN LET
Y=X-70
5540 IF X>=100 AND X<180 THEN LE
T Y=-0.25*X+55
5545 IF X>=180 THEN LET Y=X-170
5550 PLOT X+5,52+ABS(Y-50): PLO
T X+5,48-ABS(Y-50)
5555 NEXT X
5560 PRINT AT 9,10;"R"
5565 PAUSE 0
5570 CLS
5575 GO SUB 6000
5580 PRINT AT 1,0;"Realizam graf
icul relatiei  $|y|=f(x)$ "
5585 IF X=1 THEN PRINT AT 1,0;"R"
5590 FOR X=1 TO 245
5595 IF X<80 THEN LET Y=90-X
5600 IF X>=80 AND X<100 THEN LET
Y=X-70
5605 IF X>=100 AND X<180 THEN LE
T Y=-0.25*X+55
5610 IF X>=180 THEN LET Y=X-170
5615 PLOT X+5,52+ABS(Y-50): PLO
T X+5,48-ABS(Y-50)
5620 NEXT X
5625 PRINT AT 9,10;"R"
5630 PAUSE 0
5635 RETURN

```

## FUNCTII MODUL

### 1. Grafice de functii si relatii

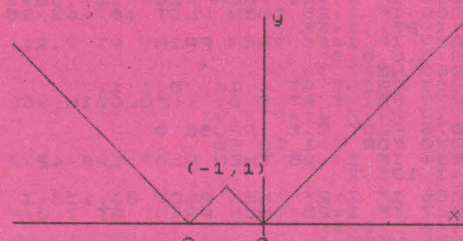


PENTRU A CONTINUA TASTATI ORICE

### 2. GRAFICUL FUNCTIEI

$$f(x) = |x+1| - 1$$

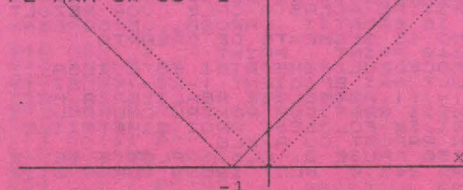
#### 2.1 MAI INTII, TRASAREA DIRECTA



PENTRU A CONTINUA APASATI ORICE TASTA

#### 2.2 FOLOSIRII PROCEDEULUI TRANSLATIEI PENTRU TRASAREA PE ETAPE A GRAFICULUI

- i) SE REALIZEAZA GRAFICUL FUNCTIEI  $f(x) = |x|$
- ii) SE TRANSALTEAZA ACEST GRAFIC PE AXA OX CU -1





## 18.41. Program TRIGRAF pentru studiul și trasarea funcțiilor

**TEMA:** Funcțiile trigonometrice  $\sin(nx)$ ,  $\cos(nx)$ ,  $\operatorname{tg}(nx)$  și graficele lor.

Scopul lecției: Studiul funcțiilor trigonometrice  $\sin(nx)$ ,  $\cos(nx)$  și  $\operatorname{tg}(nx)$ , respectiv deducerea grafică a unor formule.

**Desfășurarea lecției:** Se consideră funcțiile  $\sin: \mathbb{R} \rightarrow \mathbb{R}$ ,  $\cos: \mathbb{R} \rightarrow \mathbb{R}$  și restricțiile funcțiilor  $\sin: [0, 2\pi] \rightarrow [-1, 1]$  și  $\cos: [0, 2\pi] \rightarrow [-1, 1]$

Se demonstrează că funcțiile  $\sin$  și  $\cos$  au perioada fundamentală  $2\pi$ , justificându-se astfel alegerea domeniului de definiție  $[0, 2\pi]$ , acesta fiind suficient pentru studiul propus.

Se demonstrează  $\sin^2 t + \cos^2 t = 1$ ,  $t \in \mathbb{R}$

Se ilustrează grafic formulele:

$$\cos\left(\frac{\pi}{2} - x\right) = \sin x,$$

$$\sin\left(\frac{\pi}{2} - x\right) = \cos x.$$

Se introduc noțiunile de funcție pară, funcție impară și se arată grafic că  $\sin x$  este impară și  $\cos x$  este pară:

$$\sin(-x) = -\sin x$$

$$\cos(-x) = \cos x$$

Se găsesc intervalele de monotonie și semnul funcțiilor  $\sin x$  și  $\cos x$ .

În final se trasează graficele funcțiilor, permițându-se studiul comparativ al graficelor mixate după cum urmează:

1)  $\sin x$ ,  $\cos x$ ,  $\operatorname{tg} x$  (funcții diferite de același argument)

2)  $\sin x$ ,  $\sin 2x$ ,  $\sin 3x$

$\cos x$ ,  $\cos 2x$ ,  $\cos 3x$

$\operatorname{tg} x$ ,  $\operatorname{tg} 2x$ ,  $\operatorname{tg} 3x$  (aceleași funcții cu argumente diferite)

Rolul calculatorului este acela de a trasa graficele necesare ilustrării formulilor și teoremelor, dar și de a permite mixarea graficelor pe ecran pentru studiul comparativ.

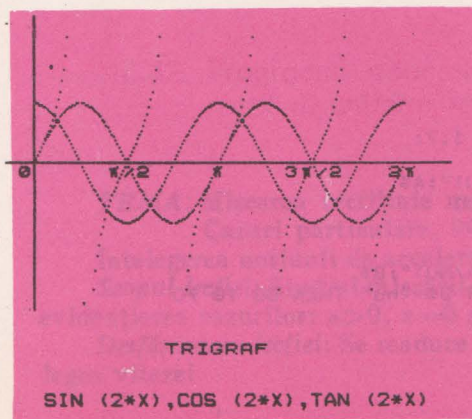
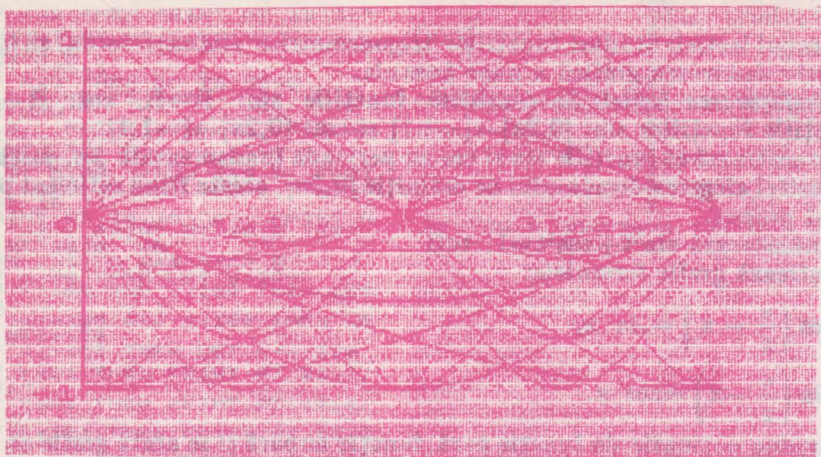


Fig. 18.3. Exemplu de imagine — ecran pentru funcții trasate cu programul TRIGRAF



Programul TRIGRAF, pe care îl prezentăm în continuare, debutează cu această imagine:



```

10 DATA 0,2,60,24,20,20,0
20 FOR I=0 TO 7: READ V: BEEP .02,I
30 POKE USR "F"+I,V: NEXT I
40 BORDER 1: PAPER 5: INK 0: CLS
50 PRINT AT 6,5:"GRAFICE TRIGONOMETRICE": FLASH 1:AT 9,12:"TRIGRAF"
55 PRINT ""TAB 10:"ELABORAT DE""TAB 6:"MIHAI ANDREI MARSANU"
56 PRINT ""TAB 17:"BUCURESTI,1987"
60 LET B$="NU": LET D=.5
70 INPUT "SIN,COS,TAN":F$: REM aceste functii se tasteaza folosind
tastele specifice functiilor dorite (in modul E )
80 PRINT AT 21,0:"FUNCTIA ESTE S$":F$+CHR$ 8;"(N*X) S=+1/-1"
90 INPUT "INTRODUCETI N si S : ";N;" ";S
100 IF S*N=0 THEN GO TO 90
110 LET S=SGN S: IF S=-1 THEN LET F$="-"+F$
120 IF N=1 THEN LET G$="(X)": GO TO 150
130 IF N=-1 THEN LET G$="(-X)": GO TO 150
140 LET G$=STR$ N: LET G$="("+G$+"*X)"
150 IF B$="NU" OR B$="nu" THEN GO SUB 330
160 PRINT FLASH 1:AT 0,11:F$+G$
170 LET F$=F$+"U"
180 FOR X=1 TO 200
190 LET U=N*X*PI/100
200 IF COS U=0 THEN GO TO 240
210 LET Y=75*VAL F$
220 IF ABS Y>75 THEN GO TO 240
230 PLOT X+25,Y+88
240 NEXT X: PRINT AT 0,0,,
250 BEEP D,0: BEEP D,2: BEEP D,4: BEEP 1,7:
BEEP D,5: BEEP D,3: BEEP D,0
260 INPUT "MAI DORITI UN GRAFIC ?(DA/NU)":A$
270 IF A$="DA" OR A$="da" THEN GO TO 300
280 IF A$="NU" OR A$="nu" THEN GO TO 400
290 GO TO 260
300 INPUT "GRAFIC ANTERIOARE RAMIN ?(DA/NU)":B$
310 IF B$="DA" OR B$="NU" OR B$="da" OR B$="nu" THEN GO TO 70
320 GO TO 300
330 CLS
340 PLOT 10,88: DRAW 240,0
350 PLOT 25,9: DRAW 0,166
360 PRINT AT 20,1:"-1":AT 1,1:"+1"

```



```

370 PRINT AT 11,2;0;AT 11,8;"P/2":REM se tasteaza P in modul grafic
380 PRINT AT 11,15;"P";AT 11,20;"3P/2"
390 PRINT AT 11,27;"2P": RETURN
400 STOP

```

#### COMENTARII LA PROGRAMUL T R I G R A F

```

10   Introducerea datelor pentru ...
20- 30 ... definirea caracterului grafic PI
40   Initializarea culorilor
50- 56 Afisarea antetului
60   Initializarea unor variabile
70   Introducerea (de la tastatura) a functiei dorite
80- 90 Introducerea (de la tastatura) a semnului functiei, si
      respectiv a coeficientului lui X
100  Testarea datelor introduse ( S si N sint diferite de zero )
110  Modificarea sirului functiei corespunzator semnului S
120-140 Formarea sirului G$ corespunzator coeficientului N
150  Testarea conditiilor de reinitializare a ecranului :
      daca B$="nu" atunci se reinitializeaza ecranul ,
      daca B$="da" atunci nu se sterge ecranul devenind astfel
      posibila mixarea graficelor
160  Afisarea formei functiei
170  Formarea sirului functiei cu argumentul U
180  Se deschide ciclul de desenare punct cu punct a functiei
190  Calculul argumentului functiei (U) in radiani tinind cont de
      numarul de puncte (200) atasat graficului
200  Daca COS U=0 rezulta TAN U nu este definita
      Pentru evitarea intreruperii programului cu mesajul respectiv
      se efectueaza testul de la linia 200
210  Se calculeaza valoarea functiei pentru argumentul U calculat
220  Daca valoarea respectiva nu este reprezentabila in spatiul
      finit al ecranului se transfera controlul liniei 240 pentru
      reluarea ciclului
230  Se deseneaza un punct de coordonate calculate anterior
      tinindu-se cont de pozitia originii axelor graficului
240  Se reia ciclul pina la terminarea trasarii graficului
      functiei pe intervalul 0,2PI
250  Semnalizare sonora
260-320 Realizarea interactiva a dialogului cu utilizatorul
330-390 Subrutina de (re)initializare a ecranului. Include:
      stergerea ecranului ,trasarea si marcarea axelor
400  Sfirsitul programului

```

### 18.42. Programul educațional ACCO pentru studiul mișcării rectilinii uniform variate

**TEMA:** Mișcarea rectilinie uniform variată.

Cazuri particulare.

Înțelegerea noțiunii de accelerație.

**Scopul lecției:** Studiul mișcării rectilinii uniform accelerate a unui corp, cu evidențierea cazurilor:  $a > 0$ ,  $a = 0$  și  $a < 0$ .

**Desfășurarea lecției:** Se readuce în discuție definiția accelerației, se deduce legea vitezei



$$v = v_0 + a \cdot t$$

(1)

și se prezintă graficul acestei legi în cazurile  $a > 0$  și  $a < 0$ .

Utilizând expresia vitezei medii se deduce legea mișcării uniform variate (legea spațiului):

$$x = x_0 + v_0 t + \frac{at^2}{2}$$

Se discută graficul legii în cele două cazuri  $a > 0$  și  $a < 0$ .

Se prezintă cazurile particulare  $x_0 = 0$  și  $v_0 = 0$ .

În final, prin eliminarea timpului din cele două ecuații se deduce formula lui Galilei:

$$v = v_0 + 2a(x - x_0) \quad (3)$$

Utilizarea calculatorului poate avea loc în mai multe momente în cadrul lecției, vizînd:

- înțelegerea noțiunii de accelerație
- ilustrarea graficului legii vitezei
- ilustrarea graficului legii mișcării uniform variate
- ilustrarea graficului vitezei în funcție de spațiu.

Pentru înțelegerea noțiunii de accelerație prezentăm programul scris în BASIC—HC-85 și comentariile aferente implementînd elementele a) și d) de mai sus.

Programul ACCO (anularea accelerației) ofera un joc educativ cu largi posibilități de extindere.

Elevul va constata, datorită acestui program educativ, ce condiții trebuie să fie îndeplinite pentru oprirea mobilului ( $v = 0$  la un anumit moment este o condiție insuficientă, fiind necesar  $a = 0$ , deci  $dv/dt = 0$  și  $v = \text{constant} = 0$ ).

Accelerația poate să ia în cadrul acestui program trei valori ( $-1$ ,  $0$  și  $+1$ ), astfel încît legile de mai sus sînt valabile pe acele intervale în care  $a = \text{constant}$ .

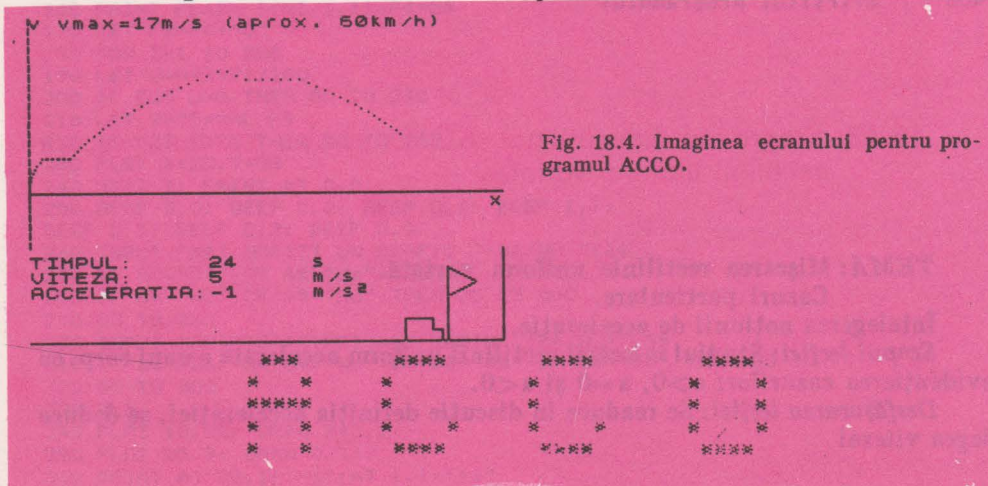


Fig. 18.4. Imaginea ecranului pentru programul ACCO.



```

10 GO SUB 350
20 IF INKEY$<>S$ AND INKEY$<>T$ THEN GO TO 20
30 OVER 1: GO SUB 620
40 GO SUB 660
50 LET A$=INKEY$
60 LET A1=(A$=S$)
70 LET A2=(A$=T$)
80 LET A=A1-A2
90 LET V=V+A*DT
100 GO SUB 620
110 FOR d=0 TO 10: NEXT d
120 LET X=X+V*DT
130 LET T=(INT (10*(T+DT)))/10
140 LET W=INT V: OVER 0
150 PRINT AT y,12;T;" "
160 PRINT AT y+1,12;W;" "
170 PRINT AT y+2,12;A;" "
180 LET M=(W=0)*(A=0)
190 LET N=(X>0)*(X<=255-L)
200 IF N=0 THEN GO TO 270
210 IF M=0 THEN GO TO 30
220 IF X<225 THEN GO TO 300
230 GO SUB 620
240 PRINT INK 2;AT 0,2;" BRAVO ! ATI OPRIT CORECT !": BEEP 2,
245 IF INKEY$<>" " THEN GO TO 245
250 INPUT "MAI INCERCATI O DATA?(D/N)";X$
260 IF X$="N" OR X$="n" THEN GO TO 999
262 IF X$<>"D" AND X$<>"d" THEN GO TO 250
265 INPUT "DORITI SA REVEDETI TEXTUL?(D/N)";X$
266 IF X$="N" OR X$="n" THEN GO SUB 440: GO TO 20
267 IF X$="D" OR X$="d" THEN RUN
268 GO TO 265
270 BEEP 1,-12: CLS
280 IF INKEY$<>" " THEN GO TO 280
285 PRINT "ATI DEPASIT SPATIUL DE DEPLASARE"
290 PAUSE 200: CLS : GO SUB 440: GO TO 20
300 FOR I=1 TO 10
310 BEEP .1,0: BEEP .1,4
320 NEXT I: CLS
330 PRINT " ATI OPRIT MOBILUL PREA REPEDE !"
340 IF INKEY$<>" " THEN GO TO 340
345 PAUSE 200: CLS : GO SUB 440: GO TO 20
350 BORDER 0: PAPER 0: INK 7: OVER 1: CLS
360 PRINT INK 2;" A C C O - PROGRAM EDUCATIV"/" PENTRU
STUDIUL MISCARII"/" RECTILINII UNIFORM VARIATE"
370 INK 5: PRINT "/" REGULILE DE BAZA ALE JOCLUI:"
380 PRINT "/" MOBILUL SE DEPLASEAZA INAINTE SI EVENTUAL INAPOI,DAR
TREBUIE OPRIT NUMAI INTRE STEAG SI ZIDUL/MARCAT PRIN MARGINEA DIN
DREAPTA A ECRANULUI."
390 PRINT "/ VA PROFUNEM SA VERIFICATI CA LA a=0 si v=0 MOBILUL
SE OPRESTEIAR PENTRU a=0 si v=0 MISCAREA CONTINUA !"
400 PRINT INK 2;"TAB 9;"ACCELERATIE 0"/"TAB 7;"PENTRU A CONTINUA"
"/
APASATI ORICE TASTA": PRINT OVER 1;AT 15,12;"/": PAUSE 0
410 CLS: GO SUB 700
420 INPUT "APASATI TASTA SELECTATA PENTRU ACCELERARE SI APOI
TASTA CR ";S$

```

## 18. PROGRAME CLASELE IX—XII



```

430 INPUT "SI ACUM ALEGETI TASTA PENTRU DECELERARE (NU UITATI
CR ! ) ";T$
440 LET X=0: LET V=1: LET T=0: LET A=0
450 LET DT=.4: LET L=20: LET H=12: LET Y=16
460 CLS: FLOT 0,0: DRAW 255,0: DRAW 0,50
470 PLOT 0,80: DRAW 255,0
480 PLOT 0,50: DRAW 0,125
490 FOR I=50 TO 175 STEP 5
500 PLOT OVER I;0,I: NEXT I
510 PLOT 225,0: DRAW 0,40
520 DRAW 15,-8: DRAW -15,-8
530 PRINT AT 12,31;"x"
540 PRINT AT 0,0;"v vmax= 17 m/s ( aprox.60 km/h )"
550 PRINT AT Y,0;"TIMPUL:";AT Y+1,0;"VITEZA:";AT Y+2,0;"ACCELERATIA:"
560 RESTORE
570 DATA 120,8,120,64,120,0,0,0,0,0,0,0,0,64,160,64
580 FOR I=0 TO 15: READ Q
590 POKE USR "a"+I,0:NEXT I
600 PRINT AT Y,19;"s";AT Y+1,19;"m/s";AT Y+2,19;"m/s "
610 RETURN
620 PLOT X,0: DRAW 0,11
630 DRAW 3*L/4,0: DRAW 0,-H/2
640 DRAW L/4,0: DRAW 0,-H/2
650 RETURN
660 OVER 0
670 PLOT X,80+5*V
680 OVER 1: RETURN
700 PRINT "" PROGRAMUL ACCO SE BAZEAZA PE UTILIZAREA LEGILOR
VITEZEI SI SFATIULUI IN MISCAREA RECTILINIEUNIFORM VARIATA:"
710 PRINT "TAB 11;" v=v0+at"
720 PRINT "TAB 8;" x=x0+";: POKE USR "b"+4,2: PRINT "v0t+at2/2":
POKE USR "b"+4,0
730 PRINT "" Prin eliminarea timpului din cele doua ecuatii
se obtine formula lui Galilei:""TAB 7;"v =(v0) +2a(x-x0)"
740 PRINT ""Acceleratia poate sa ia valorile 1,0 si -1, astfel incit
legile de mai sus sint valabile pe acele intervale in care a=ct."
750 PRINT "TAB 5;"APASATI ORICE TASTA !"
760 PAUSE 0: CLS
770 PRINT AT 5,0;" Deplasati mobilul intr-un timp
cit mai scurt pina la oprirea corecta,actionind
tastele alese pentru acceleratie/deceleratie !"
780 PRINT "" Timpul record obtinut pentru deplasarea si oprirea
corecta a mobilului este de 27 secunde !"TAB 8;"VA URAM SUCCES !"
785 PRINT FLASH 1;" MIHAI ANDREI MARSANU.
790 RETURN
999 INK 5: PAPER 1: BORDER 0: CLS

```

## COMENTARIU LA PROGRAMUL EDUCATIONAL ACCO

- 10 Apelul subrutinei de prezentare a programului și inițializare
- 20 Testarea tastei apăsată
- 30 Apelul subrutinei de DESENARE/ștergere a mobilului
- 40 Apelul subrutinei de trasare a graficului
- 50—70 A1 și A2 sînt valori de adevăr ale propozițiilor logice:  
„mobilul a fost accelerat” și „mobilul a fost decelerat”
- 80 Se calculează accelerația A (pe baza valorilor lui A1 și A2)
- 90 Inițializarea vitezei



- 100 Apelul subrutinei de desenare/ȘTERGERE a mobilului  
 110 Bucla de întârziere pentru sincronizarea cu timpul real  
 120 Inițializarea spațiului (coordonata mișcării)  
 130 Introducerea variabilei care consemnează scurgerea timpului  
 140 Calculul valorii aproximative (trunchiate) a vitezei  
 150—170 Se afișează ultimele valori calculate pentru timp, viteza și accelerație  
 180 M este valoarea de adevăr a propoziției logice:  
 „mobilul s-a oprit” care este echivalenta cu  
 „viteza este zero” și „accelerația este zero”  
 190 N este valoarea de adevăr a propoziției logice:  
 „mobilul se află în spațiul de deplasare” care este echivalentă cu  
 „coordonata mobilului este mai mare sau egală cu zero” și  
 „coordonata mobilului este mai mică decât 255-L”; unde  
 cu L am notat lungimea mobilului (mașinii)  
 200—210 Testarea lui M și N  
 220 În caz de oprire se testează poziția mobilului față de steag  
 230—240 Dacă mobilul a fost oprit corect se afișează un mesaj  
 însoțit de o semnalizare sonoră  
 245 Pauză pentru studierea imaginii finale  
 250—268 Realizarea dialogului cu utilizatorul  
 270—290 Semnalizarea depășirii spațiului de deplasare și reinițializarea auto-  
 mată a programului  
 300—345 Semnalizarea poziției incorecte de oprire (înainte de steag) și reiniți-  
 alizarea automată a programului  
 350—400 Afișarea primei pagini de prezentare  
 410 Apelul subrutinei de afișare a celei de a doua  
 pagini de prezentare  
 420—430 Selectarea tastelor pentru accelerare și respectiv decelerare (varia-  
 bilele S și respectiv T )  
 440—450 Inițializarea variabilelor (x, v, t, a ,dt, l, h, y)  
 460 Desenarea drumului și a zidului  
 470—480 Trasarea axelor graficului  $v=f(x)$   
 490—500 Marcarea axei vitezei  
 510—520 Figurarea steagului  
 530—550 Afișarea unor informații cu caracter permanent  
 560 Inițializarea pointer-ului DATA  
 570 Introducerea datelor pentru...  
 580—590 ...definirea unor caractere grafice (<sup>2</sup>, o)  
 600 Stabilirea unităților de măsură  
 610 Revenirea la programul principal  
 620—650 Subrutina pentru desenarea/ștergerea mobilului (mașinii)  
 660—680 Subrutina de trasare punct cu punct a graficului  $v=f(x)$   
 700—750 Afișarea principalelor idei și formule fizice  
 760 Temporizare  
 770—790 Afișarea paginii nr. 3 de text  
 999 Sfârșitul programului.



## MICROBIBLIOTECĂ DE PROGRAME PE CASETE

### Capitolul 19.

### Prezentarea casetelor cu programe pentru calculatorul HC-85\*

#### 19.1 Sinteza conținutului casetelor

Cărțile de față sînt însoțite (o parte a tirajului) de trei casete magnetice audio, care au înregistrări nu de... muzică, ci de programe care „interacționează” cu calculatorul personal HC-85 și cu cele compatibile.

Este vorba de 15 „micropachete de programe” destinate: învățării interactive a operării cu calculatorul HC-85 și a formării deprinderilor de a lucra la tastatură, cunoașterii configurației în care lucrează HC-85, a unității sale centrale și a proiectării circuitelor logice, studierii limbajelor BASIC și LOGO, aplicării programării în BASIC și LOGO pe exemple tipice de matematică, fizică, electronică, radioamatorism, cum și de instruire asistată și chiar de jocuri.

Programele esențiale sînt accesibile tuturor categoriilor de utilizatori: există, însă, programe orientate pe pregătire gimnazială, liceală sau universitară. Cele mai multe programe au fost scrise în BASIC STANDARD HC-85 (SINCLAIR-SPECTRUM), al cărui interpretor este rezident în memoria EPROM a calculatorului HC-85.

Interpretorul LOGO, care constituie o adaptare pentru HC 85 a interpretorului LOGO SOLI LCS1, se încarcă de pe casetă (micropachetul 4) pentru a permite lucrul cu aplicațiile în LOGO (micropachetul 5 și altele, imaginare de utilizatori). Două micropachete (7 și 8) sînt scrise în BETA BASIC (un BASIC mai performant decît cel standard); interpretoarele BETA BASIC se încarcă tot de pe casetă (din micropachetele 7, 8), pentru a permite derularea aplicațiilor respective și a altora.

Principalele micropachete în BASIC (2, 3, 9) au fost realizate la Catedra de calculatoare a Facultății de Automatică din Institutul Politehnic București, sub coordonarea Prof. dr. ing. Adrian Petrescu și a șefului de lucrări dr. ing. Nicolae Tăpus. (Lecții de BASIC — ing. Banto Ștefan\*\*. Funcționarea unui calculator la nivelul unității centrale — ing. Bărbulescu Lucian\*\*. Proiectarea cu circuite logice — ing. Păstăle Mihaela\*\*). Programul de prezentare a calculatorului HC-85 (1) a fost elaborat la Întreprinderea de Calculatoare Electronice (coordonator ing. Traian Mihu, director tehnic). Adaptarea interpretorului LOGO și aplicațiile în LOGO (4, 5) aparțin dr. ing. Nicolae Tăpus. Micropachetele 6 și 11, elaborate în BASIC, aparțin Prof. emerit Gh. Rîzescu și unor elevi ai săi din laboratorul de informatică de la Liceul Industrial Dimitrie Cantemir din București. Alte programe (14, 15) sînt elaborate de un elev de la Liceul Mihai Viteazul din București, cu asistență din partea ITCL. În sfîrșit, elevi din cercurile școlare de matematică (școala generală 30 București, cu asistență din IPB) și din cercurile de informatică ale elevilor din Buzău, Iași, Arad — asistat de CTCE Arad — au realizat micropachetele 7, 8, respectiv 10, 12, 13. Elevii autori sînt menționați la fiecare program în parte.

\* Și calculatoare compatibile cu HC-85.

\*\* Pînă în vara lui 1987, încă studenți la Facultatea Automatică.



În tabelele ce urmează sînt prezentate în ordine toate cele 15 micropachete de programe, specificîndu-se numărul, numele programului și al secțiunilor sale, lungimea (în octeți) pe care o va ocupa fiecare în memoria calculatorului, adresa de încărcare și timpul ('. ") necesar încărcării de pe casetă în memoria calculatorului.

Menționăm că limbajele BASIC HC-85 și LOGO sînt tratate pe larg în volumul 1 al cărții, cu numeroase exemplificări și programe în ambele volume, iar instrucțiunile specifice BETA BASIC (suplimentare față de BASIC HC-85) sînt date în ANEXE (vol. 2).

Mai subliniem că cele mai multe din micropachetele de programe sînt conversaționale; de asemenea, este de remarcă că la scrierea multora s-au folosit, alături de limbajele de nivel înalt (BASIC, LOGO, BETA BASIC), segmente scrise în limbaj de asamblare — cod mașină, cum și tehnici felurite, cum ar fi a modularizării, a mențurilor a editării în culori (pe monitoare color), sonorizării, graficii dinamice, instruirii prin testare, jocurilor.

### Caseta I. Micropachetele de programe 1—2—3

Nr. pachet	Tip	Nume program	Lungime — octeți —	Adresa	Timp
1.	P	hardware	4229	9100	
	C	hard	300	32256	1'20"
	C	romscr	6912	32766	
	P	T1	8431	9100	
	C	MCODE	340	32256	1'05"
	P	T2	7398	9100	
	C	mcode 2	340	32256	0'55"
	P	T3	7727	9100	
	C	mcode 2	340	32256	0'55"
	P	T4	7426	9100	
	C	mcode 2	340	32256	0'55"
2.	P	Inv CPU	16102	1	1'30"
	C	cod	6927	56985	0'35" 2'05"
	P	basie 1	8124	9900	
	C	mcode	340	32256	0'55"
	P	basie 2	8124	9900	
	C	mcode	340	32256	0'55"
	P	modul 2	5018	9900	0'30" 1'25"
	P	basie 3	7842	9900	
	C	mcode	340	32256	0'53"
	P	basie 4	8033	9900	
3.	C	mcode	340	32256	0'55"
	P	modul 4	6887	9900	0'42" 1'37"
	P	basie 5	7739	9900	
	C	mcode	340	32256	0'55"
	P	modul 5	6876	9900	0'45" 1'40"
	P	basie 6	6841	9900	
	C	mcode	340	32256	0'55"
	P	modul 6	5005	9900	0'35" 1'30"
	P	basie 7	7821	9900	
	C	mcode	340	32256	0'55"
	P	basie 8	7769	9900	
	C	mcode	340	32256	0'55"
	P	modul 8	5248	9900	0'35" 1'30"



Asistența tehnică pentru pregătirea și realizarea casetelor a fost asigurată de ing. *Eugen Dobrovie*, de la Întreprinderea de Calculatoare Electronice și de ing. *Elena Rugină* de la Electrecord, în colaborare cu colectivul de autori și cu redacția.

Pentru încărcarea unuia din cele 15 micropachete (după ce se instalează configurația calculator-casetofon-display, iar pe ecranul display-ului apare mesajul de „pregătit pentru dialog” se tastează comanda:

LOAD „nume program”, unde nume program este cel marcat cu litere îngroșate. Dacă se plasează banda casetei chiar înaintea începutului unui micropachet este suficientă comanda

LOAD ” ”

Celelalte secțiuni ale micropachetului se încarcă în mod automat sub controlul secțiunii deja încărcate.

## Caseta II. Micropachetele de programe 4—5—6—7—8

Nr. pachet	Tip	Nume program	Lungime — octeți —	Adresa	Timp
4.	P	<b>logo</b>	204	1	0'04''
	C	logoscr	6912	32768	0'38''
	C	logocode	25000	24832	2'30''
	P	logoctr 1	191	10	0'03'' 3'15''
5.	C	<b>exemple.LOG</b>	128		
	H	(7 module)	128 * 7	49722	0'16''
	C	<b>robot.LOG</b>	128	49722	
	H	(6 module)	128 * 6		0'14''
	C	<b>peisaj.LOG</b>	128	49722	
	H	(5 module)	128 * 5		0'12''
	C	<b>nim.LOG</b>	128	49722	
	H	(8 module)	128 * 8		0'18''
	C	<b>hanoi.LOG</b>	128	49722	
	H	(3 module)	128 * 3		0'08''
	C	<b>conv.LOG</b>	128	49722	
	H	(3 module)	128 * 3		0'08''
	C	<b>numără.LOG</b>	128	49722	
	H	(2 module)	128 * 2		0'06''
	C	<b>arbore.LOG</b>	128	49722	
					0'02''
	C	<b>dr.LOG</b>	128		
	H		128	49722	0'04''
	C	<b>convert.LOG</b>	128	49722	
	H	(5 module)	128 * 5		0'12''
	C	<b>sortare.LOG</b>	128	49722	
	H	(4 module)	128 * 4		0'10''
	C	<b>desen.LOG</b>	128		
	H	(12 module)	128 * 12	49722	0'26''
6.	P	<b>modul</b>	14256	5	1'20''
7.	P	<b>puteri</b>	12157	2070	1'10''
	C	bcode	18577	46960	1'54'' 3'04''
8.	C	<b>fizica</b>	21294	4080	2'00''
	C	bcode	18577	46960	1'54'' 3'54''



### Caseta III. Micropachetele de programe 9—10—11—12—13—14—15

Nr. pachet	Tip	Nume program	Lungime — octeți —	Adresa	Timp
9.	P	CLS	39998	9980	3'50"
	C	PART 3	168	65368	
	D	PART 4	205	61887	
	D	PART 5	463	62095	
10.	P	morse	8295	1	0'44"
11.	P	conice	39621	1	3'35"
12.	P	ROMANIA	827	10	
	C	1	192	65200	
	C	2	6912	40000	3'24"
	C	3	6912	50000	
	P	4	10041	1	
13.	P	MLS	154	0	0'03"
	P	MLSO	35134	3	3'10"
	C	M/C	3535		0'20" 3'33"
14.	P	trigraf	3007	1	20"
15.	P	acco	4029	1	25" 45"

Alte informații de încărcare se dau în paragrafele 19.2—19.4.

În prima coloană se află numărul micropachetului de programe (1—15). În a doua coloană se află înscris simbolul tipului de program: P — program BASIC; C — program în cod mașină; A — bloc de date; H — secțiune de program, fără header, fără nume, care se încarcă sub LOGO.

Majoritatea micropachetelor de programe este compusă din mai multe secțiuni, cu denumiri în coloana a treia, a numelui program, prima dintre denumiri, dată cu litere îngroșate (aldine), reprezentând programul BASIC.

Menționăm că aplicațiile în LOGO (micropachetul 5) se pot încărca și lansa în execuție numai după ce s-a încărcat interpretorul LOGO (micropachetul 4). O aplicație în LOGO se încarcă prin comanda LOAD „nume aplicație”, fără a mai fi necesară specificarea extensiei. LOG

În coloana a 4-a se găsesc specificate lungimile (în octeți) ocupate de fiecare secțiune în memorie. În coloana a 5-a se specifică adresa de încărcare în memorie a secțiunilor; pentru programele BASIC se dă numărul de linie al primei secțiuni ce se va executa; pentru programele în cod mașină — adresa fizică de memorie unde va începe încărcarea secțiunii; pentru programele în LOGO — adresa de început a memoriei tampon asociate interpretorului LOGO.

În ultima coloană sînt date duratele secvențelor, secțiunilor sau / și ale întregului micropachet de programe.

### 19. PREZENTAREA PROGRAMELOR



## 19.2 Caseta 1. Prezentarea și încărcarea programelor

**1. Denumirea programului.** Prezentarea calculatorului HC-85 și învățarea tastaturii.

**Prezentarea programului.** Programul constă din două părți.

Prima parte prezintă structura, componentele și modul de operare ale calculatorului HC-85.

Partea a doua conține patru lecții de învățare a tastaturii: modul de lucru tip mașină de scris, cuvintele cheie, modul cursorului, modul extins.

**Modul de utilizare.** Programul are prin excelență un caracter conversațional.

Generează mesaje de pornire/oprire a casetofonului. Fiecare lecție se termină cu o secțiune de teste, care vin să verifice și să consolideze cunoștințele și deprinderile însușite în privința operării la tastatură.

După poziționarea corectă a benzii, înainte de programul pe care îl dorim să-l încărcăm se execută o comandă de încărcare.

Astfel pentru încărcarea primei părți se introduce LOAD „hardware“, iar pentru încărcarea programelor din partea a doua

LOAD „T1” — pentru lecția 1 ... LOAD „T4” — pentru lecția 4

Dacă banda este poziționată chiar înaintea unuia dintre aceste programe, pe zona nelăregistrată, atunci încărcarea programului se realizează prin comanda: LOAD ” ”

**2. Denumirea programului:** Funcționarea unui calculator la nivelul unității centrale.

**Prezentarea programului.** Programul are ca scop prezentarea elementelor funcționale ce intră în componența unității centrale: unitatea aritmetică-logică (UAL), unitatea de comandă (UC) și unitatea de memorie internă (UM).

Sînt ilustrate grafic etapele de citire/interpretare și execuție ale instrucțiunilor, la nivelul resurselor fizice-funcționale: ceas, registre, magistrale, UAL, UC și UM.

**Modul de utilizare.** Programul este conversațional, utilizatorul asigurînd numai trecerea de la o secvență la alta prin apăsarea unei taste.

Încărcarea programului se realizează prin execuția comenzii: LOAD „inv. CPU”, banda fiind poziționată înaintea programului. Dacă poziționarea este chiar pe zona nelăregistrată dinaintea programului încărcarea se poate face și cu comanda LOAD ” ”

Programele sînt conversaționale, utilizatorul trebuind să răspundă la diversele mesaje pe care le afișează calculatorul.

Menționăm că la ultimul mesaj afișat de calculator utilizatorul nu mai poate încărcă partea de exemplu, care a fost eliminată de pe casetă. Se trece, deci, la 3.

**3. Denumirea programului.** Lecții de BASIC.

**Prezentarea programului.** Programul conține 8 lecții destinate învățării într-o manieră conversațională a limbajului BASIC.

Lecția 1: are un caracter introductiv, prezintă noțiunile de bază în programare.

Lecția 2: constante, variabile, operații aritmetice și cu siruri.

Lecția 3: linii-comentarii, instrucțiuni de atribuire și de intrare.

Lecția 4: attribute ecran, instrucțiuni de tipărire. (are și continuare)

Lecția 5: instrucțiuni de salt, decizii, condiții, cicluri, subrutine.

Lecția 6: funcții (are și continuare).

Lecția 7: set caractere grafice, sunet.

Lecția 8: memoria; scrierea programelor, folosirea casetofonului.

**Modul de utilizare:** Programele sînt conversaționale, utilizatorul trebuind să răspundă la diversele mesaje pe care le afișează calculatorul.

Fiecare lecție este compusă dintr-o parte de program și o parte de cod mașină, iar unele dintre ele mai au încă o parte de program.

Încărcarea acestor lecții se realizează prin încărcarea primei părți de program BASIC asociat fiecărei lecții. Celelalte părți (cod mașină sau program) asociate fiecărei lecții se încarcă sub controlul programului, utilizatorul nefiind nevoit să le încarce în mod explicit.

**IX. MICROBIBLIOTECA PE CASETE**



Astfel pentru încărcarea lecțiilor trebuie executată una dintre comenzi:

LOAD "basic 1" pentru lecția 1... LOAD "basic 8" pentru lecția 8

Dacă banda este poziționată chiar înaintea unuia dintre aceste programe, atunci încărcarea programului respectiv se realizează prin comanda LOAD " "

## 19.3 Caseta II. Prezentarea și încărcarea programelor

### 4. Denumirea programului: Interpretor LOGO

**Prezentarea programului:** Programul conține interpretorul LOGO care asigură mediul de dezvoltare programe în limbajul LOGO.

După încărcarea programului se intră în regim de comandă, în care utilizatorul poate introduce și executa programe LOGO.

Interpretorul pune la dispoziția utilizatorului un editor prin care acesta poate să-și elaboreze programul sursă ce urmează să fie interpretat și executat.

Primitivele limbajului LOGO sînt descrise în partea a V-a în cadrul capitolului 11.

În continuare, pe casetă, se află un set de programe LOGO a căror descriere se găsește în capitolul 12. Încărcarea acestora se face sub controlul interpretorului LOGO utilizînd comanda LOAD "nume" (vezi paragraf 11.9.1), iar lansarea în execuție se face prin numele procedurii urmat de parametrul asociați acesteia.

Programul constituie o adaptare pe calculatorul HC-85 a interpretorului pentru limbajul LOGO elaborat de LOGO SOLI / LCS1.

**Modul de utilizare:** Pentru elaborarea și executarea unui program scris în limbajul LOGO este necesar să se încarce mai întîi interpretorul LOGO. Acest lucru se realizează prin execuția comenzii: LOAD "logo"

sau dacă caseta este poziționată chiar înaintea interpretorului (la începutul casetei 2) prin execuția comenzii LOAD " "

După încărcarea interpretorului, apare prompterul care ne invită să introducem primitive ale limbajului LOGO

### 5. Denumirea programului: Aplicații în LOGO

**Prezentarea programului:** Pentru înțelegerea programelor scrise în limbajul LOGO și care se găsesc pe casetă este necesar să se consulte capitolul 12, care conține o descriere detaliată a fiecăruia, atît ca funcționare, ca parametri de intrare și mod de apelare.

Aplicațiile LOGO sînt:

exemple. LOG par. 12.3.1 pag. 298; robot. LOG — 12.1 pag. 278; peisaj. LOG — 12.1 pag. 279; nim. LOG — 12.1 pag. 280; hanoi. LOG — 12.3.7 pag. 309; conv. LOG — 12.3.4 pag. 303; numără. LOG — 12.2 pag. 289; arbore. LOG — 12.2 pag. 293; dr. LOG — 12.2 pag. 291; convert. LOG — 12.3.5 pag. 304; sortare. LOG — 12.3.4 pag. 302; desen. LOG — 12.3.6 pag. 307

**Mod de utilizare:**

Se poziționează caseta cît mai aproape de programul care se dorește să se încarce chiar înaintea lui, și se execută comanda LOAD "numeprogram"

De exemplu pentru încărcarea programului LOGO, peisaj. LOG se execută comanda: LOAD "peisaj"

În timpul încărcării apar pe ecran numele procedurilor definite în cadrul programului. Pentru lansare în execuție se introduce numele programului: peisaj.

În cazul programelor care au și parametri de intrare, pentru execuție trebuie specificat numele urmat de parametrii corespunzători: LOAD "arbore" arbore 40



## 6. Denumirea programului: Modul

**Prezentarea programului\*.** Programul se adresează în primul rând elevilor din clasele IX—XII și are ca scop să ajute la dezvoltarea cunoștințelor despre funcția modul și a deprinderilor elevilor în utilizarea sa. Sub o formă grafică adecvată sînt trecute în revistă o serie de proprietăți ale funcției și relației modul. Plecînd apoi de la o funcție martor prezentată inițial și menținută pe parcursul folosirii modulului sînt prezentate o serie de funcții și relații care folosesc modulul.

### Modul de utilizare:

Programul se încarcă prin execuția comenzii `LOAD'' modul''`, sau a comenzii `LOAD ''`, dacă caseta a fost poziționată chiar înainte programului modul.

Comanda `RUN` (linia) evidențiază noi funcții sau relații, de ex. `RUN 20`:

Linia 20— Grafice de funcții și relații  $y = |x|$ ;  $y = -|x|$ ;  $|y| = x$ ;  $|y| = |x|$

Linia 190 — Graficul funcției  $f(x) = ||x+1|-1|$  (trasare directă)

Linia 350 — Folosirea procedurii translației pentru trasarea pe etape a graficului

Linia 670 — Realizarea graficului funcției  $f: \mathbb{R} \rightarrow \mathbb{R}$ :

$f(x) = x-4$ ,  $x \leq -2$ ;  $x$ ,  $-2 < x \leq -1$ ;  $-1/4 x - 5/4$ ,  $-1 < x \leq 3$ ;  $x-5$ ,  $x > 3$

Linia 700 — realizează graficul funcției  $y = |f(x)|$

Linia 946 — realizează graficul funcției  $y = f(|x|)$ , după care urmează subrutinele pentru graficele funcțiilor și relațiilor următoare:  $y = -f(|x|)$ ,  $y = |f(|x|)|$ ,  $y = -|f(|x|)|$ ,  $y = |f(-|x|)|$ ,  $y = -|f(-|x|)|$ ,  $|y| = f(x)$ ,  $|y| = |f(-|x|)|$

Linia 950 — Graficul relației  $|2y-1| + |2y+1| + 4\sqrt[3]{3}|x|-4=0$

Linia 1120 — Graficul relației  $||x| + |y| - 3| - 3| = 1$

## 7. Denumirea programului: puteri\*\*

**Prezentarea programului:** programul se adresează elevilor din clasa a V-a și are ca scop operațiile elementare cu puteri naturale.

Sub o formă grafică adecvată, care include și mișcare, programul prezintă:

— înmulțirea și împărțirea de puteri cu aceeași bază;

— puterea unui produs și a unei puteri.

Ca aplicație la cele prezentate se calculează ultima cifră a rezultatului ridicării unui număr natural la o putere reprezentată printr-un număr natural ( $n^m$ ), unde  $0 \leq n$ ,  $m < 99999999$ . La cerere, programul poate prezenta modul în care a făcut calculul. De menționat faptul că deși rezultatul  $n^m$  nu poate fi reprezentat în calculator pentru  $n$  și  $m$  foarte mari se calculează ultima cifră.

În finalul programului se propune un test care verifică însușirea cunoștințelor.

Programul a fost scris în *BETABASIC*, folosindu-se facilitățile de protecție sistematică a programelor oferite de acest interpretor. Programul este modular, realizîndu-se proceduri pentru diverse secțiuni ale algoritmului. Pentru a fi executat este necesar să se încarce codul interpretorului *BETA BASIC* (bcode) de pe casetă în memoria calculatorului.

**Modul de utilizare:** după poziționarea corectă a casetei, înainte de programul puteri, se execută comanda `LOAD ''puteri''` sau `LOAD ''` (dacă s-a poziționat chiar pe zona nelăregistrată dinaintea programului). Secvența de cod bcode este încărcată de către program, după care se lansează în execuție.

Pentru secțiunea de prezentare a operațiilor, programul se derulează automat, fără intervenția utilizatorului. Pentru aplicație, utilizatorul specifică baza și exponentul.

În cadrul testului de la sfîrșit, utilizatorul trebuie să dea răspunsul. În cazul unui răspuns eronat calculatorul arată cum trebuie rezolvat testul. De subliniat faptul că s-a implementat un editor la nivel de linie pentru introducerea răspunsului.

\* Programul „MODUL” a fost realizat de prof. emerit Gh. Rîzescu și de elevul său Bălan Radu (anul 1986).

\*\* Realizator: elevul Tăpuș Cristian, clasa a VI-a, Școala generală nr. 30 București.



## 8. Denumirea programului: fizica\*\*

**Prezentarea programului:** programul se adresează elevilor de clasa a VI-a și prezintă electrizarea corpurilor. Sub o formă grafică adecvată, care include și mișcare, se prezintă aspecte privind: electrizarea prin frecare, prin contact, prin influență, interacțiunea corpurilor electrizate.

În final, programul implementează un joc bazat pe interacțiunea sarcinilor pozitive și negative. Pe suprafața ecranului se generează aleator, ca poziție, un număr de maxim 5 ioni pozitivi (+) și negativi (-). Unui electron (o), aflat în colțul din stînga jos, i se imprimă o mișcare într-o anumită direcție. Mișcarea electronului este influențată de sarcinile aflate pe ecran. *Programul a fost scris în Beta Basic, în aceleași condiții ca la 7.*

**Modul de utilizare:** după poziționarea corectă a casetei, înainte de programul fizica, se execută comanda LOAD "fizica" sau LOAD "" (dacă s-a poziționat chiar pe zona nel înregistrată dinaintea programului). *Secvența de cod bcode este încărcată de către program, după care se lansează în execuție.* Pe baza unui meniu, utilizatorul poate să mute cursorul prin intermediul tastelor A (jos) și Q (sus) și să aleagă toate problemele legate de electrizare sau o anumită problemă individuală. Trecerea de la o secvență ecran la alta se realizează prin intermediul apăsării unei taste.

Pentru jocul propus, programul oferă instrucțiuni de folosire. După stabilirea numărului de jucători și a nivelului de dificultate pe ecran apar ioni pozitivi (+) și negativi (-) care vor interacționa cu electronul (o). Electronul poate primi un impuls inițial de direcție stabilită prin tastele Q (sus), A (jos), O (stînga), P (dreapta) care pot muta un cursor pe marginea ecranului. Direcția inițială este stabilită de linia imaginată care unește electronul (o) cu cursorul.

După stabilirea direcției, electronul începe să se deplaseze și este atras și respins de sarcinile de pe suprafața ecranului conform legii lui Coulomb. Jucătorul primește un număr de puncte proporțional cu produsul  $x * y$ , unde  $x$  și  $y$  reprezintă coordonatele punctului unde electronul întilnește marginea ecranului,

## 19.4 Caseta III. Prezentarea și încărcarea programelor

### 9. Denumirea programului. Proiectarea cu circuite logice.

**Prezentarea programului.** Programul prezintă în cadrul a trei etape proiectarea schemelor cu circuite (porți) logice de tip ȘI (AND), SAU (OR), NU (NOT), ȘI-NU (NAND), SAU-NU (NOR).

Prima parte tratează structura generală a unui calculator și tipurile de porți logice de bază.

Partea a doua ilustrează diverse posibilități de interconectare a porților logice, pentru a forma circuite mai complexe. Sînt exemplificate modalitățile de simulare a funcționării unui tip de poartă, printr-o combinație de alte porți logice, precum și funcționarea unui circuit logic secvențial, utilizat frecvent în calculatoare pentru a realiza funcția de memorie.

Partea a treia oferă posibilitatea de a proiecta circuite proprii cu un număr de maximum 12 porți logice, avînd 4 intrări și 4 ieșiri.

Programul asigură conectarea porților și furnizează informații referitoare la eventualele erori, oferind facilitățile necesare pentru corectare. După validarea circuitului, se poate urmări funcționarea, dînd diverse valori pentru mărimea de intrare.

Schema proiectată poate fi afișată la imprimantă.

**Modul de utilizare:** Programul conversațional, este o adaptare a lui "Make a clip".

Trecerea de la o secvență la alta se realizează prin apăsarea unei taste. Pentru proiectarea unui circuit propriu, utilizatorul dispune de un meniu care li asigură interfața cu calculatorul.

Încărcarea programului se realizează prin execuția comenzii: LOAD "CLS" sau dacă banda este poziționată chiar înaintea programului CLS prin LOAD ""

Dacă programul este oprit, din diferite motive, se poate relansa cu comanda GOTO 1.

De menționat faptul că acest program se execută numai după casetă. Dacă se încarcă de pe disc, nu se poate executa partea a treia.

\*\* Realizator: elevul Țăpuș Cristian, clasa a VI-a, Școala generală nr. 30 București.



### 10. Denumirea programului: „MORSE”\*

**Prezentare:** Programul execută transmiterea de semnale telegrafice cu impulsuri de la 10 la peste 400 semne pe minut, afișind pe ecran grupe de 5 × 5 semnale morse transmise. Se pot comanda până la 50 grupe a 5 caractere aleator, combinat sau pe grupe, așa cum prevăd „manualul radioamatorului” și normele IARU de învățare a alfabetului Morse și pentru lucrul în concursuri.

În program mai sînt două subrutine mari care transformă tastatura în mașină de transmis texte codificate sau „în clar” prin alfabetul Morse, util sistemului de transmitere S × TTY din practica radioamatorilor. Programul a fost probat pe „modem × TTY” și lucrează pe cele trei frecvențe standard.

**Utilizare:** Programul se încarcă cu comanda LOAD „Morse”

### 11. Denumirea programului: conice\*\*

**Prezentarea programului:** Programul se adresează elevilor din clasa a XI-a și are ca scop abordarea capitolului „Conice” din programa de geometrie analitică. Sub o formă grafică adecvată se prezintă proprietățile conicelor.

**Modul de utilizare:** După poziționarea corectă a casetei, înainte de programul conice, se execută comanda LOAD „conice”

sau dacă banda este poziționată chiar înainte de program (pe zona nelăregistrată dinaintea programului) pentru încărcare se execută LOAD „ ”

Programul este interactiv, utilizatorul putînd să aleagă diversele aspecte referitoare la conice

### 12. Denumirea programului: ROMANIA, județele țării\*\*\*

**Prezentarea:** Program de instruire asistată de calculator în domeniul geografiei; realizat în mai multe variante: pentru HC-85 și Sinclair Spectrum 48, pentru HC-85 cu disc flexibil, pentru Sinclair ZX Spectrum + 2(128 K). Scris în BASIC și cod mașină, condus prin meniu, poate fi adaptat și pentru alte teme decît figurarea pe hărți sau cu alte date geografice-istorice, mai complexe.

**Utilizarea:** Introduceți LOAD „ ” (sau LOAD\*„d”; 1; „ROMANIA” pentru versiunea de pe disc); restul programului se încarcă automat, apare meniul; cu tasta S se poziționează pe opțiunea dorită; lansarea în execuție cu ENTER; la opțiunea a 2-a județele se introduc cu majuscule; se citește tastatura, se apelează rutinele; cele în cod mașină salvează și readuc în RAM hărțile ROMÂNIEI, mai există subrutine de prezentare grafică și sonoră; programul rezident-33 KO.

Sînt de menționat artificii de programare: utilizarea ferestrelor, caractere grafice definite de utilizator, încărcător în cod mașină, efectul de „dizolvare” a ecranului, memorarea unui ecran și reafășarea instantanee.

### 13. Denumirea programului: Roboțelul MINITEHNICUS la școală\*\*\*\*

**Prezentare:** Program de instruire asistată și jocuri pe calculator, scris în BASIC, cu rutine în limbajul de asamblare Z 80, schimbare a caracterelor grafice, sonorizare, cu folosirea unor programe utilitare ca Melbourne Draw, Gens 3m, Mons 3m; ocupă 46 KO. Roboțelul este condus de operator în cabinetele de matematică, fizică, istorie, geografie, informatică, răspunde la un set total de ~100 întrebări afișate aleator; la 5 răspunsuri corecte pe cabinet îl poate părăsi primește un punctaj și o melodie „clopotel”; cînd greșește— punctajul scade și roboțelul nu înaintează; cînd are 350 puncte poate intra în ultimul cabinet, unde joacă un joc „TIR SPAȚIAL”. Întrebări de nivelul clasei a VI-a, ce pot fi înlocuite cu alt set de

\* Realizator: elevul Pipoiu Iulian, clasa a VII-a, cercul de informatică din Buzău.

\*\* Program realizat de prof. emerit Gh. N. Rîzescu împreună cu elevul său Tudor Florin

\*\*\* Realizator: elevul Răzvan Petrescu clasa a VII-a, Liceul „Costache Negruzzi”, (1986) din Iași.

\*\*\*\* Realizator: elevul Răzvan Jigorea, clasa a VI-a, Școala generală nr 5, (1986) din Arad și CTCE Arad.



pe caseta magnetică, cu instrucțiunea MERGE. Premiat cu marele premiu — la concursul republican de informatică pentru elevi „Năvodari '86” și cu un premiu special — la concursul internațional de la SOFIA 1987. Programul este achiziționat de ITCI București, prin contract de cercetare.

**Utilizare:** Introducți LOAD "mls"; urmează un dialog interactiv operator-calculator

#### **14. Denumirea programului: TRIGRAF\***

**Prezentarea programului:** Programul permite trasarea și mixarea funcțiilor trigonometrice  $\sin(NX)$ ,  $\cos(NX)$  și  $\operatorname{tg}(NX)$ .

TRIGRAF este comentat pe larg în subcapitolele 16.5.3. (tema nr. 9) și 18.3.1.

**Modul de utilizare:** Programul se încarcă cu instrucțiunea LOAD " " după poziționarea corectă a benzii.

#### **15. Denumirea programului: ACCO\***

**Prezentarea programului:** ACCO este un program — joc educativ, care permite studiul mișcării rectilinii uniforme variate, facilitând înțelegerea noțiunii de accelerație (condiția necesară pentru oprirea unui mobil fiind  $V=0$  și  $a=0$ ).

ACCO este comentat pe larg în subcapitolele 16.5.3. (tema nr 11) și 18.42.

**Modul de utilizare:** Programul se încarcă cu instrucțiunea LOAD ",," după poziționarea corectă a benzii.

### **INDICAȚII PENTRU LUCRUL CU DISCHETE**

Pentru HC 85 extins (cu disc flexibil) programele de pe casete se pot trece pe disc fie prin utilizarea unui program t2d, fie încărcând de pe casete cu LOAD „nume program” în memorie și salvind pe disc cu comanda „SAVE \* "d"; 1;” nume program”.

Ulterior programele de pe dischete se pot încărea pentru execuție cu comanda LOAD \* "d"; 1; \* "nume program” atât pe HC-85 extins cit și pe HC-88. (A se vedea și textul PROLOG... DIALOG... EPILOG din vol. 1 și 2).

---

\* Realizator: elevul Mihai Andrei Mârșanu, clasa a XII-a, Liceul Mihai Viteazul, București (1986).



## COMPLEMENTE DE MATEMATICĂ\*

### Capitolul 20. | Sisteme de numerație poziționale

#### 20.1. Noțiuni introductive

Există mai multe sisteme de numerație poziționale:

1°. Sistemul zecimal, în care oamenii numără și socotesc în mod obișnuit. Acest sistem folosește pentru reprezentarea numerelor zece simboluri elementare numite cifre, ele aparținând mulțimii

$\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

2°. Sistemul binar, care folosește pentru reprezentarea numerelor cifrele 0 și 1.

3°. Sistemul ternar care folosește cifrele 0, 1 și 2.

4°. Sistemul octal, ale cărui cifre aparțin mulțimii

$\{0, 1, 2, 3, 4, 5, 6, 7\}$

5°. Sistemul hexazecimal (sau „hexal”), ale cărui șaisprezece simboluri sînt elemente ale mulțimii  $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$ .

Lista acestor sisteme de numerație nu se încheie aici.

Există și sisteme de numerație nepoziționale, spre exemplu — sistemul de numerație roman în baza zece.

#### 1.1. Valoare nominală

Se observă că fiecare din sistemele enumerate utilizează un anumit număr de simboluri și fiecare simbol are o valoare nominală a sa, de exemplu: de la 0 la 9 pentru sistemul zecimal și de la 0 la 7 pentru cel octal, de la 0 la F pentru cel hexazecimal etc.

*Prin definiție*, numărul natural egal cu numărul cifrelor utilizate într-un sistem de numerație se numește baza sistemului.

Sistemul zecimal are baza zece, cel binar — baza 2, ..., cel hexazecimal — baza șaisprezece.

Baza oricărui sistem de numerație pozițional se reprezintă simbolic prin „10”. Ea este numită unitatea de ordinul întâi (sau de rang unu). Simbolurile elementare ale bazei sînt unități de ordinul zero (sau de rang zero); 10 unități de ordinul întâi formează o unitate de ordinul doi — reprezentată simbolic prin „100”; 10 unități de ordinul doi formează o unitate de ordinul trei — reprezentată simbolic prin „1000” etc.

\* ...Lucrări folosite în scrierea acestui capitol:

Ioana Bărbat, Alexandru Dumitrache; *Matematica aplicată în tehnica de calcul clasa a IX-a*, E.D.P., București, 1982.

I. Săndulescu, D. Beloiu *Introducere în informatică*, clasa a IX-a, E.D.P., București, 1978.



Evident, în sistemul binar, care are doar simbolurile 0 și 1, două unități de ordinul zero formează o unitate de ordinul întâi, reprezentată prin „10”. În cel ternar — trei unități de ordinul zero formează o unitate de ordinul întâi, reprezentată prin „10”, ..., în cel hexal — șaisprezece unități de ordinul zero formează o unitate de ordinul întâi, reprezentată simbolic tot prin „10” etc.

În tabelul 20.1 se poate observa corespondența dintre reprezentările unor numere în sisteme de numerație cu diverse baze „b”.

Tabelul 20.1

Reprezentarea numerelor în diverse baze de numerație

Sistem zecimal b=10	Sistem binar b=2	Sistem octal b=8	Sistem hexal b=16	Sistem zecimal b=10	Sistem binar b=2	Sistem octal b=8	Sistem hexal b=16
0	0	0	0	17	10001	21	11
1	1	1	1	18	10010	22	12
2	10	2	2	19	10011	23	13
3	11	3	3	20	10100	24	14
4	100	4	4	21	10101	25	15
5	101	5	5	22	10110	26	16
6	110	6	6	23	10111	27	17
7	111	7	7	24	11000	30	18
8	1000	10	8	25	11001	31	19
9	1001	11	9	26	11010	32	1A
10	1010	12	A	27	11011	33	1B
11	1011	13	B	28	11100	34	1C
12	1100	14	C	29	11101	35	1D
13	1101	15	D	30	11110	36	1E
14	1110	16	E	31	11111	37	1F
15	1111	17	F	32	100000	40	20
16	10000	20	10				

## Valoare pozițională

Fie cifra 8 din sistemul zecimal. Dacă în fața numărului 8 așezăm cifra 9, aceasta va reprezenta cifra zecilor a numărului 98 din sistemul zecimal. Așadar, simbolurile sistemului zecimal, avînd fiecare o valoare nominală de la 0 la 9, au și o valoare pozițională — corespunzătoare ordinului: zecilor, sutelor, miilor, zecilor de mii, sutelor de mii, milioanele etc. Desigur, simbolurile tuturor sistemelor de numerație poziționale au atît valoare nominală cît și valoare pozițională.

## Scrierea numărului cu ajutorul puterilor bazei

Fie numărul zecimal 256. El conține două unități de ordinul doi (ordinul sutelor), cinci unități de ordinul întâi (ordinul zecilor) și șase unități de ordinul zero (ordinul unităților). Valorile nominale și poziționale ale acestui număr sînt puse în evidență cu ajutorul tabeli 20.2.

Se observă ușor că numărul 256 poate fi reprezentat cu ajutorul puterilor bazei sub forma unui polinom aritmetic.

$256 = 2 \cdot 10^2 + 5 \cdot 10^1 + 6 \cdot 10^0$ , unde  $10^0 = 1$ , iar  $10^1$  se scrie 10 și rezultă reprezentarea  $256 = 2 \cdot 10^2 + 5 \cdot 10 + 6$ .

În acest mod, orice număr de trei cifre — conținînd  $a_2$  unități de ordinul doi,  $a_1$  unități de ordinul întâi și  $a_0$  unități de ordinul zero, se reprezintă sub forma pozițională prin alăturarea unităților sale:  $a_2 a_1 a_0$ , iar sub forma polinomială (cu ajutorul puterilor bazei)  $a_2 \cdot 10^2 + a_1 \cdot 10 + a_0$ . Deci  $a_2 a_1 a_0 = a_2 \cdot 10^2 + a_1 \cdot 10 + a_0$ .

## 20. NUMERAȚIE POZIȚIONALĂ



Tabelul 20.2

milioane	sute de mii	zeci de mii	mii	sute	zeci	unități
$10^6$	$10^5$	$10^4$	$10^3$	$10^2$	$10^1$	$10^0$
1000000	100000	10000	1000	100	10	1
$0 \times 1000000$	$0 \times 100000$	$0 \times 10000$	$0 \times 1000$	$2 \times 100$	$5 \times 10$	$6 \times 1$
0	0	0	0	2	5	6

Generalizînd, se poate spune că un număr natural oarecare  $N$ , avînd  $a_n$  unități de ordinul  $n$ ,  $a_{n-1}$  unități de ordinul  $(n-1)$ , ...,  $a_1$  unități de ordinul întâi și  $a_0$  unități de ordinul zero, se poate reprezenta convențional sub formă pozițională — prin alăturarea coeficienților  $a_i$  și sub formă polinomială — după puterile descrescătoare ale bazei „10”. Avem deci scrierea pozițională

$$N = a_n a_{n-1} a_{n-2} \dots a_2 a_1 a_0$$

și scrierea polinomială — cu ajutorul puterilor bazei

$$N = a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + a_{n-2} \cdot 10^{n-2} + \dots + a_2 \cdot 10^2 + a_1 \cdot 10 + a_0$$

unde  $0 \leq a_i \leq 9$ ,  $i \in \{0, 1, 2, \dots, n-1\}$ ,  $a_n > 0$

Scrierea numerelor sub această formă poate fi extinsă. Astfel, un număr pozitiv  $N$  care are și parte întreagă și parte fracționară se poate scrie sub forma pozițională

$$N = a_n a_{n-1} a_{n-2} \dots a_2 a_1 a_0, a_{-1} a_{-2} \dots a_{-k},$$

iar cu ajutorul puterilor bazei, sub forma polinomială

$$N = a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + a_{n-2} \cdot 10^{n-2} + \dots + a_1 \cdot 10 + a_0 + a_{-1} \cdot 10^{-1} + a_{-2} \cdot 10^{-2} + \dots + a_{-k} \cdot 10^{-k}$$

Concret, numărul 1987, 2578 se scrie:

$$1987,2578 = 1 \cdot 10^3 + 9 \cdot 10^2 + 8 \cdot 10 + 7 + 2 \cdot 10^{-1} + 5 \cdot 10^{-2} + 7 \cdot 10^{-3} + 8 \cdot 10^{-4}$$

В десятичной системе счисления каждая цифра имеет свое место. Например, в числе 1987,2578 цифра 1 стоит в разряде тысяч, 9 — в разряде сотен, 8 — в разряде десятков, 7 — в разряде единиц, 2 — в разряде десятых, 5 — в разряде сотых, 7 — в разряде тысячных, 8 — в разряде десятитысячных.

$$x = \pm a_n a_{n-1} a_{n-2} a_{n-3} \dots a_2 a_1 a_0, a_{-1} a_{-2} \dots a_{-k} \dots,$$

iar scrierea sa sub formă polinomială, cu ajutorul puterilor bazei, de expresia

$$x = \pm (a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + a_{n-2} \cdot 10^{n-2} + \dots + a_2 \cdot 10^2 + a_1 \cdot 10 + a_0 + a_{-1} \cdot 10^{-1} + a_{-2} \cdot 10^{-2} + \dots + a_{-k} \cdot 10^{-k} + \dots)$$

Dacă numărul de cifre de după virgulă ale numărului real  $x$  este prea mare (sau infinit), atunci se poate înlocui  $x$  printr-o aproximantă a sa, notată  $x^*$  — acest număr avînd un număr de zecimale determinat.

## 20.2. Sistemul de numerație binar

Simbolurile elementare ale sistemului binar sînt 0 și 1. Numărului 2 din sistemul pozițional zecimal îi corespunde 10 în sistemul pozițional binar, lui 4 îi corespunde 100, lui 8 îi corespunde 1000, ..., lui 64 îi corespunde 1000000 etc.

Datorită unor facilități de ordin tehnic sistemul binar a fost generalizat pentru informatică și calculator (procesoarele sau microprocesoarele folosind informația binară). Într-adevăr, celor două cifre ale sistemului binar li se pot pune în corespondență cele două stări de magnetizare obținute la trecerea curentului printr-un circuit electronic (magnetizării Nord-



Sud corespunzându-i simbolic cifra „1“, iar celei Sud-Nord—obținută la schimbarea sensului curentului—cifra „0“. Aceeași corespondență cu cifrele „1“ și „0“ ale sistemului binar se realizează și în cazul stărilor de tensiune (Înaltă „1“ sau joasă „0“), în cazul stărilor unui tranzistor de a conduce sau de a fi blocat, în cazul celor două stări ale unui contact bipozițional (Închis sau deschis), ..... , în cazul valorilor adevărat-fals ale logicii bivalente a algebrei booleene.

Pentru tehnica electronică de calcul semnul cifric „0“ sau „1“, din sistemul binar, este numit bit\* și constituie unitatea de măsură a informației în calculator. O unitate de măsură mai mare, formată din opt biți, se numește octet sau bait („byte“).

Un octet poate reprezenta orice număr de la 0 la 255 (în binar „11111111“); un grup de doi octeți (16 biți) alcătuiește un semicuvânt și reprezintă orice număr de la 0 la 65535 (în binar „1111 1111 11111111“); un grup de patru octeți formează un cuvânt (32 biți). Cuvântul ca unitate de măsură servește la scrierea datelor și a instrucțiunilor în minicalculator. Și în sistemul binar, simbolurile 0 și 1 au o valoare nominală și una pozițională, valoarea pozițională fiind determinată de puterea la care se găsește baza 2. În tabelul 20.3 este dată reprezentarea binară a numărului 367.

Tabelul 20.3

2 <sup>8</sup>	2 <sup>7</sup>	2 <sup>6</sup>	2 <sup>5</sup>	2 <sup>4</sup>	2 <sup>3</sup>	2 <sup>2</sup>	2 <sup>1</sup>	2 <sup>0</sup>
256	128	64	32	16	8	4	2	1
1.256	0.128	1.64	1.32	0.16	1.8	1.4	1.2	1.1
1	0	1	1	0	1	1	1	1

Se observă că numărul 367<sub>10</sub> (367 în baza 10) se reprezintă cu ajutorul puterilor bazei 2 astfel:

$$1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 256 + 64 + 32 + 8 + 4 + 2 + 1 = 367$$

## Numărarea în binar

Tehnica numărării în binar este similară celei din zecimal, unde zece unități simple (ordinul unităților) formează o unitate de ordin superior (ordinul zecilor) etc. În acest sens numerația în baza 2 este prezentată în tabelul 20.4. Conform tablei, două unități simple formează o unitate de ordin superior, adică 10; adăugând o unitate se obține 11 și încă una — se obține 100 (deci 4<sub>10</sub>=100<sub>2</sub>) etc.

Tabelul 20.4

Sistem zecimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Sistem binar	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110
Sistem zecimal	15	16	17	18	19	20	21	22	23	24					
Sistem binar	1111	10000	10001	10010	10011	10100	10101	10110	10111	11000					
Sistem zecimal	25	26	27	28	29	30	31	32	33	34					
Sistem binar	11001	11010	11011	11100	11101	11110	11111	100000	100001	100010					

\* „Bit“ provine din contracția euvintelor din limba engleză „binary digit“ (cifră binară).



## Conversia zecimal-binar

Echipamentele periferice ale calculatorului transformă numerele scrise în zecimal în echivalentul lor binar. Această transformare (conversie) se realizează prin împărțiri succesive ale numărului dat în sistemul zecimal la baza 2 a sistemului binar, pînă ce se obține restul 0 sau 1, calculul făcîndu-se în baza zece. Spre exemplu, pentru conversia în baza 2 a numărului zecimal 367, șirul operațiilor de împărțire la bază constă în parcurgerea următorilor pași :

$$367:2=183 \text{ (rest 1 notat „r}_0\text{“),}$$

$$183:2 = 91 \text{ (rest 1, notat „r}_1\text{“),}$$

91:2 45 (rest 1, notat „r<sub>2</sub>)“,

$$45:2 = 22 \text{ (rest 1, notat „r}_3\text{“),}$$

$$22:2=11 \text{ (rest 0, notat „r“),}$$

$$11:2 = 5 \text{ (rest 1, notat „r}_5\text{“),}$$

$$5:2 = 2 \text{ (rest 1, notat „r}_6\text{“),}$$

$$2:2 = 1 \text{ (rest 0, notat „r}_7\text{“),}$$

$$1:2=0 \text{ (rest 1, notat „r}_8\text{“)}$$

În final, se citesc cifrele restului împărțirilor succesive, de la sfârșit către început obținându-se astfel numărul în baza 2:

$$\Gamma_8 \quad \Gamma_7 \quad \Gamma_6 \quad \Gamma_5 \quad \Gamma_4 \quad \Gamma_3 \quad \Gamma_2 \quad \Gamma_1 \quad \Gamma_0$$

$$\text{Deci } 367_{(10)} = 101101111_{(2)}$$

Din exemplul prezentat se constată că algoritmul pentru trecerea unui număr natural  $N$  din baza 10 în baza 2 constă în următorii pași:

$$\frac{N}{2} = k_1, \text{ rest } r_0; \frac{k_1}{2} = k_2, \text{ rest } r_1 \dots; \frac{k_{q-1}}{2} = k_q, \text{ rest } r_{q-1}; \frac{k_q}{2} = k_{q+1}, \text{ rest } r_q.$$

Ultima împărțire efectuată este cea pentru care  $k = 1$ , numărul  $N$  scriindu-se în baza 2 sub forma „ $r_q r_{q-1} r_{q-2} \dots r_2 r_1 r_0$ ”.

Algoritmul descris poate fi folosit în mod practic fără a se întrerupe pașii împărțirii. Astfel, resturile obținute prin împărțirea succesivă la bază, încadrate în pătrate, sînt transcrise în final — în ordinea inversă apariției lor (tabelul 20.5)

**Tabelul 20.5**

Diagram illustrating the construction of the Sierpinski triangle using the iterated function system (IFS) approach. The diagram shows the sequence of points  $z_n$  and the corresponding transformations  $T_i$  applied to the previous point  $z_{n-1}$ .

The points  $z_n$  are labeled as follows:

- $z_0 = 0$
- $z_1 = 1$
- $z_2 = 2$
- $z_3 = 3$
- $z_4 = 4$
- $z_5 = 5$
- $z_6 = 6$
- $z_7 = 7$

The transformations  $T_i$  are labeled as follows:

- $T_1$  (from  $z_0$  to  $z_1$ )
- $T_2$  (from  $z_1$  to  $z_2$ )
- $T_3$  (from  $z_2$  to  $z_3$ )
- $T_4$  (from  $z_3$  to  $z_4$ )
- $T_5$  (from  $z_4$  to  $z_5$ )
- $T_6$  (from  $z_5$  to  $z_6$ )
- $T_7$  (from  $z_6$  to  $z_7$ )

The diagram shows the sequence of points  $z_n$  and the corresponding transformations  $T_i$  applied to the previous point  $z_{n-1}$ . The points  $z_n$  are labeled as follows:

- $z_0 = 0$
- $z_1 = 1$
- $z_2 = 2$
- $z_3 = 3$
- $z_4 = 4$
- $z_5 = 5$
- $z_6 = 6$
- $z_7 = 7$

The transformations  $T_i$  are labeled as follows:

- $T_1$  (from  $z_0$  to  $z_1$ )
- $T_2$  (from  $z_1$  to  $z_2$ )
- $T_3$  (from  $z_2$  to  $z_3$ )
- $T_4$  (from  $z_3$  to  $z_4$ )
- $T_5$  (from  $z_4$  to  $z_5$ )
- $T_6$  (from  $z_5$  to  $z_6$ )
- $T_7$  (from  $z_6$  to  $z_7$ )

$$\text{Deci } 255_{10} = (r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0)_2 = 1111 \ 1111_2 ; 73_{10} = 1001001_{(2)}$$

Ca aplicație să se arate că:

$$1) 65535_{(10)} = 1111 \ 1111 \ 1111 \ 1111_{(2)}$$

$$2) -359_{(10)} = -101100111_{(2)}$$



## Observații

i) Conversia zecimal-ternar sau zecimal-octal, deci conversia din baza zece într-o bază mai mică se realizează tot prin împărțiri succesive, resturile obținute aparținând mulțimii simbolurilor noii baze. Astfel, numărul  $475_{10}$  are ca echivalent binar numărul  $111011011_{(2)}$ , ca echivalent ternar numărul  $122121_{(3)}$ , iar ca echivalent octal — numărul  $733_{(8)}$ .

ii) Conversia numerelor care au și parte întreagă și parte zecimală se efectuează examinând separat, atât partea întreagă cât și partea zecimală. Astfel, fie numărul

$$x = a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + a_{n-2} \cdot 10^{n-2} + \dots + a_2 \cdot 10^2 + a_1 \cdot 10 + a_0 + a_{-1} \cdot 10^{-1} + a_{-2} \cdot 10^{-2} + \dots + a_{-k} \cdot 10^{-k},$$

a căruia parte fracționară este dată de „ $a_{-1}a_{-2} \dots a_{-k}$ ”.

Conversia părții fracționare într-o bază  $b$  se reduce la determinarea în noua bază a echivalentului acestei părți—notat prin „ $b_{-1}b_{-2} \dots b_{-k}$ ”

Echivalentul căutat se obține prin înmulțirea succesivă a părții fracționare date și a celei obținute pe parcurs cu noua bază, reținând de fiecare dată întregul astfel obținut.

Spre exemplu, echivalentul în bază 2 al numărului  $73,875_{(10)}$  este  $1001001,111_{(2)}$ . Într-adevăr,  $73_{(10)} = 1001001_{(2)}$ , iar pentru calculul echivalentului binar al numărului  $0,875_{(10)}$  sînt necesari următorii pași:

$$0,875 \cdot 2 = 1 + 0,75; \quad b_{-1} = 1$$

$$0,75 \cdot 2 = 1 + 0,5; \quad b_{-2} = 1$$

$$0,5 \cdot 2 = 1 + 0; \quad b_{-3} = 1$$

Deci,  $0,875_{(10)} = 0,111_{(2)}$  și avem transformarea:  $73,875_{(10)} = 1001001,111_{(2)}$

iii) În conversia numerelor subunitare se poate lua un număr determinat de zecimale, depinzînd de precizia cerută.

## Numere zecimale codificate binar

Dacă fiecareia din cifrele de la 0 la 9, ale codului zecimal, i se pune în corespondență o configurație de patru cifre binare (4 biți), se obține conversia numerelor zecimale (0—9) în numere binare cu patru ranguri (numite tetrade). Corespondența între cifrele zecimale și tetradele binare este dată în tab. 20.6, ea reprezentînd o codificare binară a codului zecimal. Informația ce se introduce în calculator este adeseori reprezentată într-un astfel de cod.

Deoarece cu un număr binar cu patru ranguri se pot codifica 16 cifre diferite, iar sistemul zecimal necesită numai 10 combinații, rezultă că există mai multe posibilități de realizare a unei corespondențe între cifrele zecimale și tetradele binare.

Codul bazat pe conversia dată în tabela 20.6 este un cod ponderat, ponderile avînd de la stînga la dreapta valorile 8, 4, 2, 1. Pentru acest motiv el este numit codul ponderat 8421. Codul zecimal este astfel codificat binar, această reprezentare binară cu patru ranguri a numerelor zecimale fiind numită reprezentarea binar-zecimală, sau codul binar-zecimal. Folosirea codului binar-zecimal în conversia numerelor zecimale a condus la dezvoltarea sistemului binar zecimal de reprezentare a numerelor. Astfel, pentru conversia binar zecimală a numărului  $675_{(10)}$  se înlocuiesc caracterele cifrice ale fiecărui ordin al numărului dat prin tetradele binare corespunzătoare (tabelul 20.7). O legătură între reprezentarea numerelor în cod zecimal, cod binar-natural și cod binar zecimal este dată în tabelul 20.8

Pentru efectuarea operațiilor aritmetice, în cazul unui calculator care lucrează în sistemul binar, este necesară conversia din codul binar zecimal în binar — la introducerea datelor și conversia lor în binar zecimal — la extragerea rezultatelor din calculator.



Tabelul 20.6

Tabelul 20.6 (Correspondențe între cifrele zecimale și tetradele binare)

Cifre zecimale	Binar				Tetradele binare
	$2^3=8$	$2^2=4$	$2^1=2$	$2^0=1$	
0	0	0	0	0	0 0 0 0
1	0	0	0	1	0 0 0 1
2	0	0	1	0	0 0 1 0
3	0	0	1	1	0 0 1 1
4	0	1	0	0	0 1 0 0
5	0	1	0	1	0 1 0 1
6	0	1	1	0	0 1 1 0
7	0	1	1	1	0 1 1 1
8	1	0	0	0	1 0 0 0
9	1	0	0	1	1 0 0 1

Tabelul 20.7

Tabelul 20.7 (Model de corespondență binar zecimal-zecimal)

	sute				zeci				unități			
	$2^3$	$2^2$	$2^1$	$2^0$	$2^3$	$2^2$	$2^1$	$2^0$	$2^3$	$2^2$	$2^1$	$2^0$
binar zecimal	0	1	1	0	0	1	1	1	0	1	0	1
zecimal	6				7				5			

Tabelul 20.8

Tabelul 20.8. (Reprezentarea numerelor în coduri binare).

Cod zecimal	Cod binar- natural	Cod binar- zecimal	Cod zecimal	Cod binar-natural	Cod binar-zecimal
0	0	0 0 0 0	10	1010	00010000
1	1	0 0 0 1	11	1011	00010001
2	10	0 0 1 0	12	1100	00010010
3	11	0 0 1 1	13	1101	00010011
4	100	0 1 0 0			
5	101	0 1 0 1	20	10100	00100000
6	110	0 1 1 0			
7	111	0 1 1 1	100	1100100	0001 0000 0000
8	1000	1 0 0 0			
9	1001	1 0 0 1			



## Conversia binar-zecimal

Pentru trecerea în baza zece a unui număr exprimat în baza doi, se scrie numărul binar sub formă de polinom aritmetic cu baza doi și se efectuează calculul în baza zece. Se utilizează deci formula:  $N = a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + \dots + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0 \cdot 2^0$ ,

unde:

$N$  este numărul dat în binar;  $a_i (0 \leq i \leq n)$  reprezintă caracterele 0 și 1 ale sistemului binar, iar  $n$  indică valoarea pozițională a caracterului binar

deci:

$$10001111_{(2)} = 1 \cdot 2^8 + 0 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 256 + 16 + 8 + 4 + 2 + 1 = 287_{(10)}$$

Deci echivalentul în baza zece al numărului  $10001111_{(2)}$  este 287.

Algoritmul pentru conversia numărului  $N$  din baza 2 în baza 10 este ușor de înțeles dacă se ține seama de algoritmul pentru conversia din baza 10 în baza 2. Pornind de la acest algoritm rezultă:

$$k_q = 2k_{q+1} + r_q; \quad k_{q-1} = 2k_q + r_{q-1} = 2^2 \cdot k_{q+1} + 2r_q + r_{q-1};$$

$$N = 2k_1 + r_0 = r_q \cdot 2^q + r_{q-1} \cdot 2^{q-1} + r_{q-2} \cdot 2^{q-2} + \dots + r_2 \cdot 2^2 + r_1 \cdot 2^1 + r_0.$$

Deci, conversia din baza 2 în baza 10 necesită realizarea următorilor pași:

1) Se consideră forma pozițională a numărului binar

$$r_q \ r_{q-1} \ r_{q-2} \ \dots \ r_2 r_1 r_0_{(2)}$$

2) Se trece această formă în sistemul zecimal, pe baza relației:  $r_q \cdot 2^q + r_{q-1} \cdot 2^{q-1} + \dots + r_1 \cdot 2^1 + r_0 \cdot 2^0$  cu calculul în baza zece.

$$\text{Exemplu: } 1001101_2 = 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^0 = 64 + 8 + 4 + 1 = 77_{10}$$

*Observație.* În cazul numerelor binare întregi

$$a_n a_{n-1} a_{n-2} \dots a_2 a_1 a_0, \quad a_i \in \{0,1\}, \quad i = \overline{0, n},$$

conversia în zecimal se poate realiza și pe baza relației

$$(\dots(((a_n \cdot 2 + a_{n-1}) \cdot 2 + a_{n-2}) \cdot 2 + a_{n-3}) \cdot 2 + \dots + a_1) \cdot 2 + a_0$$

Aplicînd acest algoritm numărului „ $a_n a_{n-1} a_{n-2} a_{n-3} a_{n-4} a_{n-5} a_{n-6} a_{n-7} a_{n-8} a_{n-9} a_{n-10}$ ” rezultă

$$N = (((((a_n \cdot 2 + a_{n-1}) \cdot 2 + a_{n-2}) \cdot 2 + a_{n-3}) \cdot 2 + a_{n-4}) \cdot 2 + a_{n-5}) \cdot 2 + a_{n-6}) \cdot 2 + a_{n-7}) \cdot 2 + a_{n-8}) \cdot 2 + a_{n-9}) \cdot 2 + a_{n-10}$$

În cazul numărului  $101101_2$ , rezultă  $101101_2 = (((((1 \cdot 2 + 0) \cdot 2 + 1) \cdot 2 + 1) \cdot 2 + 0) \cdot 2 + 1) \cdot 2 + 1) \cdot 2 + 0 = 45$

## Operații aritmetice în binar

Algoritmii operațiilor aritmetice în sistemele de numerație poziționale sînt independenți de baza particulară folosită, regulile de calcul fiind deci aceleași.

Regulile după care se efectuează cele patru operații aritmetice în sistemul binar sînt definite, tabelate sau exemplificate în cele ce urmează:

1°. Adunarea binară are la bază regula:

$$0+0=0$$

$$0+1=1$$

$$1+0=1$$

$1+1=10$  (transportînd „1” ordinului următor, deoarece  $1+1=10$ , deci  $1+1=0+1$  (report pentru rangul binar de la stînga).

## 20. NUMERAȚIE POZIȚIONALĂ



Tabelul 20.9

+	0	1
0	0	1
1	1	10

Tabelul 20.9—a

2	0 0 1 0 <sub>(2)</sub>
+ 7	0 1 1 1 <sub>(2)</sub>
<hr/>	<hr/>
9	1 0 0 1 <sub>(2)</sub>
în zecimal	în binar

Tabelul 20.9—b

53	1 1 0 1 0 1 <sub>(2)</sub>
+ 37	1 0 0 1 0 1 <sub>(2)</sub>
<hr/>	<hr/>
90	1 0 1 1 0 1 0 <sub>(2)</sub>
în zecimal	în binar

2°. Scăderea binară are la bază regula:

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$0 - 1 = 1 \text{ (împrumut de la ordinul următor)}$$

$$1 - 1 = 0$$

Tabelul 20.10

-	0	1
0	0	1
1	1	0

Tabelul 20.10—a

90	1 0 1 1 0 1 0 <sub>(2)</sub>
- 37	- 1 0 0 1 0 1 <sub>(2)</sub>
<hr/>	<hr/>
53	1 1 0 1 0 1 <sub>(2)</sub>
în zecimal	în binar

3°. Înmulțirea binară are la bază regula:

$$0 \cdot 0 = 0$$

$$0 \cdot 1 = 0$$

$$1 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

Pentru efectuarea înmulțirii a două numere se înmulțește de înmulțitul cu fiecare ordin al înmulțitorului, adunându-se în final produsele obținute (Tabelul 20.11).

4°. Împărțirea binară se execută ca și împărțirea în zecimal, aplicând însă regulile cunoscute la înmulțirea și scăderea binară (tabelul 20.12).

45	1 0 1 1 0 1 <sub>(2)</sub>
5	1 0 1 <sub>(2)</sub>
<hr/>	<hr/>
225	1 0 1 1 0 1 <sub>(2)</sub>
în zecimal	1 0 1 1 0 1 <sub>(2)</sub>
	<hr/>
	1 1 1 0 0 0 0 1 <sub>(2)</sub>



Tabelul 20.12

225	5	1 1 1 0	0 0 0 1	1 0 1
20	45	1 0 1		
25		1 0 0		
25		0 0 0		
00		1 0 0 0		
		1 0 1		
		0 1 1 0		
		1 0 1		
		0 0 1 0		
		0 0 0		
		0 1 0 1		
		1 0 1		
		0 0 0		

### 20.3. Sistemul de numerație octal

Simbolurile sistemului de numerație pozițional octal sînt cifrele de la 0 la 7. Corespondența între cifrele sistemului octal și cifrele celui zecimal este dată în tabelul numerotat 20.13.

Tabelul 20.13

Zecimal	1	2	3	4	5	6	7	8	9
Octal	1	2	3	4	5	6	7	10	11

Conversia zecimal-octal se realizează prin împărțiri succesive ale numărului exprimat în sistemul zecimal la baza  $b=8$ , calculul făcîndu-se în baza 10.

De exemplu:  $347_{10}=533_8$

Conversia octal-zecimal se realizează scriind numărul octal sub formă de polinom aritmetic cu baza 8 și efectuînd calculul corespunzător în baza 10.

De exemplu:  $533_8=5 \cdot 8^2+3 \cdot 8^1+3=320+24+3=347_{10}$

Un interes deosebit pentru tehnica de calcul îl prezintă conversia binar-octal și invers. Baza 8 fiind o putere a lui 2 ( $8=2^3$ ), este evidentă corespondența biunivocă între numerele binare cu trei ranguri (triade) și cifrele sistemului octal (tabelele 20.14—*a* și 20.14—*b*).

Fiecare caracter numeric al sistemului octal este reprezentat printr-un grup de trei cifre binare, numit triadă.

## 20. NUMERAȚIE POZIȚIONALĂ



Tabelul 20.14—a

Reprezentarea octal-binar

Octal	Binar		
	$2^3=4$	$2^2=2$	$2^0=1$
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Tabelul 20.14—b

Corepondența binar-octal

Binar	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Tabelul 20.14 — c. Triade

$N_2 =$	$\frac{1}{\downarrow 1}$	$\frac{101}{\downarrow 5}$	$\frac{110}{\downarrow 6}$	$\frac{001}{\downarrow 1}$	$\frac{010}{\downarrow 2}$
---------	--------------------------	----------------------------	----------------------------	----------------------------	----------------------------

Pentru conversia directă a unui număr din sistem binar în sistem octal, se grupează cifrele sale binare, luate câte trei, începând din dreapta pentru numere întregi, scriindu-se apoi echivalentul lor în octal. Spre exemplu, pentru numărul  $N_2=110111\ 000\ 1010$  se realizează gruparea în triade din tabelul 20.14—c și se obține  $N_8=15612$

Desigur conversia binar octal se poate face și prin intermediul bazei 10. De exemplu:

$$1101110_{(2)}=110_{(10)}=156_8$$

## Operații aritmetice în octal

Operațiile în octal se bazează pe reguli similare operațiilor din sistemul zecimal. Pentru adunare, scădere, înmulțire și împărțire se utilizează tabelele 20.15 (a, b, c, d).

Tabelul 20.15-a

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Tabel 20.15-b

0—0=0
1—0=1
2—1=1
3—1=1
4—2=2
5—3=2
6—4=2
7—4=3
10—4=4
etc.

Tabel 20.15-c

.	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Tabel 20.15-d

1 : 1 = 1
4 : 2 = 2
14 : 3 = 4
20 : 4 = 4
17 : 3 = 5
etc.



$34_8$	$34_8$	$34_8$	$610_8$	$34_8$
$+16_8$	$-16_8$	$\times 16_8$	$34_8$	$16_8$
$52_8$	$16_8$	$250_8$	$250_8$	
		$34_8$	$250_8$	
		$610_8$	$==$	

## 20.4. Sistemul de numerație hexazecimal

Baza sistemului hexazecimal este 16 și se notează prin „10”.

Simbolurile sistemului hexazecimal (pe scurt „hexal”) sînt cifrele și numerele de la 0 la 15, desemnate astfel: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Corespondența între caracterele sistemului zecimal și cel hexal este dată în tabelul 20.16.

Tabelul 20.16. Corespondența zecimal-hexal

Zecimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
hexal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Baza 16 fiind o putere a lui 2 ( $16=2^4$ ), exprimarea caracterelor cifrice ale sistemului de numerație hexazecimal în sistemul binar se realizează cu ajutorul grupărilor de cîte patru cifre binare din tabelul 20.16—a.

Astfel:  $0_{16}=0000_2$ ;  $1_{16}=0001_2$ ;  $2_{16}=0010$ ;  $3_{16}=0011_2$

$4_{16}=0100_2$ ; ...;  $10_{16}=1010_2$ ;  $11_{16}=1011_2$ ; ...;  $15_{16}=1111_2$ .

Tabelul 20.16 — a

Hexazecimal	binar			
	$2^3$	$2^2$	$2^1$	$2^0$
0	0	0	0	0
1	0	0	0	0
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

4.1. *Conversia zecimal hexazecimal.* Această conversie se realizează prin împărțiri succesive ale numărului exprimat în sistemul zecimal la baza  $b=16$ , calculul făcîndu-se în baza zece.

De exemplu, etapele conversiei în hexal a numărului  $984_{10}$  sînt date în figura 20.16 — b

## 20. NUMERAȚIE POZIȚIONALĂ



Tabelul 20 16-b

$$984 : 16 = 61, \text{ rest } 8$$

$$61 : 16 = 3, \text{ rest } 13 \text{ (adică D)}$$

$$3 : 16 = 0, \text{ rest } 3$$

$$\text{Deci } 984_{10} = 3D8_{16}$$

### Conversia hexazecimal – zecimal

Această conversie se realizează folosind scrierea polinomială a numărului. Astfel

$$3D8_{16} = 3 \cdot 16^2 + D \cdot 16^1 + 8 \cdot 16^0 = 768 + 208 + 8 = 984_{10}$$

### Conversia hexal-binar și invers

1°. Conversia hexal binar se realizează fie prin intermediul bazei 10, fie direct.

Spre exemplu, fie  $N_{16} = A01F$

Correspondentul acestui număr în baza zece se obține folosind mai întâi scrierea polinomială și făcând calculul în baza zece:

$$A01F_{16} = A \cdot 16^3 + 0 \cdot 16^2 + 1 \cdot 16^1 + F \cdot 16^0 = A \cdot 4096 + 16 + 15 = 40960 + 16 + 15 = 40991_{10}$$

Se realizează apoi conversia zecimal-binar prin împărțiri succesive la baza 2.

$$40991_{10} = 10100000000 11111_2$$

Conversia directă, din baza 16 în baza 2, se realizează înlocuind fiecare din simbolurile numărului hexazecimal prin echivalentul corespunzător (cu 4 ranguri) din baza 2.

$$\text{Deci } A01F_{16} = 1010 0000 0001 1111_2$$

2°. Conversia binar-hexal se realizează grupând cifrele binare ale numărului în seturi de câte patru, începând din partea dreaptă spre stînga și scriind apoi echivalentul fiecărei grupe în hexal. De exemplu,  $N_2 = 110100000000 11111 = 1 1010 0000 0001 1111 = 1A01F_{16}$

### Operații aritmetice în hexazecimal

Operațiile frecvent folosite în hexazecimal pentru programarea calculatoarelor sînt adunarea și scăderea. În acest sens sînt date cîteva modele privind efectuarea acestor operații:

$$\begin{array}{r} 2B1F_{16} \\ + DAC_{16} \\ \hline 105E9_{16} \end{array} \quad \begin{array}{r} 75A3_{16} \\ + 3A9B_{16} \\ \hline B03E_{16} \end{array} \quad \begin{array}{r} B03E_{16} \\ - 3A9B_{16} \\ \hline 75A3_{16} \end{array}$$

Pașii parcurși în rezolvarea primului exercițiu sînt:

I)  $(A+F)_{16} = 10 + 15 = 25 = 19_{16}$  (se scrie „9”, report „1”)

II)  $(1+C+1)_{16} = 1 + 12 + 1 = 14 = E_{16}$

III)  $(A+B)_{16} = 10 + 11 = 21 = 15_{16}$  (se scrie „5”, report „1”)

IV)  $(1+D+2)_{16} = 1 + 13 + 2 = 16 = 10_{16}$ .

Deci, suma căutată este  $105E9_{16}$ .

Pentru facilitarea calcului se folosesc tabelele privind adunarea și înmulțirea în hexal, pentru numere mai mici decît baza (tabelele 20.17 și 20.17-a).



+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Tabelul 20.17. Adunarea în hexal

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	0	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E
3	0	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0	7	E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	0	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	0	9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	0	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	0	B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	0	C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	0	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	0	E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	0	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

Tabelul 20.17 — a. Înmulțirea în hexal.



## 21.1. Operatorii logici de bază

În logica bivalentă, deci în logica cu două valori de adevăr, se numește propoziție\* un enunț despre care știm că este sau adevărat sau fals, însă nu și una și alta simultan.

Dacă o propoziție este adevărată, spunem că ea are valoarea de adevăr „adevărul” și vom nota valoarea de adevăr prin semnul „1” (sau „a”); dacă este falsă, spunem că ea are valoarea de adevăr „falsul” și vom nota această valoare prin semnul „0” (sau „f”).

Vom nota propozițiile cu literele  $p, q, r \dots$  sau  $p_1, p_2, p_3, \dots$

Aceste propoziții se pot compune cu ajutorul unor operatori (sau conectori) logici, dând propoziții din ce în ce mai complexe, notate prin majuscule ale alfabetului latin sau prin litere ale alfabetului grec.

În logica bivalentă există patru operatori logici unari și șaisprezece operatori logici binari. Dintre aceștia, trei sînt considerați operatori de bază (sau operatori primitivi), deoarece cu ajutorul lor se pot exprima toți ceilalți operatori ai logicii bivalente. Aceștia sînt: negația (operator unar), conjuncția și disjuncția (operatori binari).

În ordinea utilității lor urmează implicația și echivalența. Toți acești operatori sînt prezentați în manualul de algebră pentru clasa a IX-a.\*\*

Operatorii (sau conectorii) logici sînt întîlniți și sub denumirea de conexiuni logice, sau de funcții logici, sau funcții logice.

Definițiile operatorilor pot fi date și cu ajutorul unor tabele, numite tabele (sau matrice) de adevăr.

Notînd propozițiile prin literele  $p$ , respectiv  $q$ , adevărul prin simbolul „1” sau „a” și falsul prin simbolul „0” sau „f”, avem definițiile operatorilor de negație, conjuncție, disjuncție, implicație și echivalență din tabela 21.1 (a, b, c, d, e).

Tabela 21.1 (a, b, c, d, e)

## OPERATORI LOGICI

P	$\bar{P}$	P	q	$P \wedge q$	P	q	$P \vee q$	P	q	$P \rightarrow q$	P	q	$P \leftrightarrow q$
0	1	0	0	0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	0	1	1	0	1	1	0	1	0
		1	0	0	1	0	1	1	0	0	1	0	0
Negația tabela 2.1-a		1	1	1	1	1	1	1	1	1	1	1	1
		Conjuncția tabela 2.1 - b			Disjuncția tabela 2.1 - c			Implicația tabela 2.1 - d			Echivalența tabela 2.1 - e		

\* Un enunț este un asamblaj de semne cărora li s-a dat un sens.

\*\* Algebra. Manual pentru clasa a IX-a. Autori: C. Năstăsescu, C. Niță, Gh. Rizescu; E.D.P. — București, 1979.



Definițiile prezentate se pot formula în cuvinte astfel:

a) Negatia propoziției  $p$  este propoziția „non  $p$ ” (sau „nu  $p$ ”), care se notează  $\neg p$  (sau  $\bar{p}$ ) și care este adevărată cînd  $p$  este falsă și falsă cînd  $p$  este adevărată.

b) Conjunția propozițiilor  $p, q$  este propoziția compusă care se citește „ $p$  și  $q$ ”, notată „ $p \wedge q$ ”, care este adevărată atunci și numai atunci cînd fiecare din propozițiile  $p, q$  este adevărată.

c) Disjuncția propozițiilor  $p, q$  este propoziția compusă care se citește, „ $p$  sau  $q$ ”, notată „ $p \vee q$ ” și care este adevărată atunci și numai atunci cînd este adevărată cel puțin una dintre propozițiile  $p, q$ .

d) Implicația propozițiilor  $p, q$  (în această ordine) este propoziția compusă care se citește „ $p$  implică  $q$ ”, notată „ $p \rightarrow q$ ” și care este falsă atunci și numai atunci cînd  $p$  este adevărată și  $q$  falsă, în celelalte cazuri fiind adevărată.

În implicația „ $p \rightarrow q$ ”,  $p$  se numește ipoteza sau antecedentul implicației, iar  $q$  se numește concluzia sau consecventul implicației. Implicația „ $p \rightarrow q$ ” se exprimă prin formula „ $(\neg p) \vee q$ ”.

e) Echivalența propozițiilor  $p, q$  este propoziția compusă notată „ $p \leftrightarrow q$ ”, care se citește „ $p$  echivalent cu  $q$ ”, sau „ $p$  dacă și numai dacă  $q$ ”, și care este adevărată atunci și numai atunci cînd  $p, q$  sînt în același timp adevărate sau false.

Din analiza tabelelor prin care s-au definit operatorii  $\wedge, \vee, \rightarrow$  și  $\leftrightarrow$ , se observă că evaluările cuplului  $(p, q)$  s-au scris ținîndu-se seama că se cunosc mai întîi evaluările propoziției  $p$ . Deci, din dubletul  $(p, q)$  se obțin mai întîi cuplurile  $(0, *)$  și  $(1, *)$ , unde prin semnul „\*” se înțelege mai întîi „0” și apoi „1”. Se obțin astfel cuplurile  $(0, 0), (0, 1), (1, 0)$  și  $(1, 1)$  pentru evaluarea dubletului  $(p, q)$ , deci  $2^2=4$  astfel de evaluări.

În cazul unui triplet de proporții bivalente  $(p, q, r)$ , pentru fiecare din cele patru evaluări ale dubletului  $(p, q)$  se obțin cîte două evaluări pentru tripletul  $(p, q, r)$ . Spre exemplu: din evaluarea  $(0, 0)$  se obțin tripletele  $(0, 0, 0)$  și  $(0, 0, 1)$ ; din evaluarea  $(0, 1)$  — tripletele  $(0, 1, 0)$  și  $(0, 1, 1)$  etc. Rezultă că tripletul  $(p, q, r)$  are de două ori mai multe evaluări decît dubletul  $(p, q)$  și anume, elementele mulțimii.

$\{(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)\}$ .

Se observă că tripletul  $(p, q, r)$  se poate evalua în  $2^3=8$  moduri, matricea de adevăr avînd deci  $2^3$  linii. Se demonstrează că un  $n$ -uplu de propoziții bivalente  $(p_1, p_2, \dots, p_n)$  se poate evalua în  $2^n$  moduri, matricea de adevăr avînd deci  $2^n$  linii. În tabela 21.1-f este prezentată, ca exemplu, matricea proprietății de distributivitate a conjuncției față de disjuncție:

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$$

unde s-a notat:  $\alpha = p \wedge (q \vee r)$ ,  $\beta = (p \wedge q) \vee (p \wedge r)$ , iar echivalența formulelor prin semnul „ $\equiv$ ”.

Tabela 21.1 — f

$p$	$q$	$r$	$q \vee r$	$p \wedge (q \vee r)$	$p \wedge q$	$p \wedge r$	$(p \wedge q) \vee (p \wedge r)$	$\alpha \equiv \beta$
0	0	0	0	0	0	0	0	1
0	0	1	1	0	0	0	0	1
0	1	0	1	0	0	0	0	1
0	1	1	1	0	0	0	0	1
1	0	0	0	0	0	0	0	1
1	0	1	1	1	0	1	1	1
1	1	0	1	1	1	0	1	1
1	1	1	1	1	1	1	1	1

### Observații

1°. Atît pentru demonstrarea echivalenței formulelor, cît și pentru analiza expresiilor logice care au un număr mic de propoziții ( $n \leq 4$ ), se folosește în mod obișnuit procedeul matriceal.



Matricea de adevăr are  $2^n$  linii orizontale,  $n$  fiind numărul variabilelor propoziționale diferite ale expresiei logice date. Numărul 2 semnifică aici numărul valorilor logice pe care le poate lua o variabilă propozițională.

Numărul coloanelor matricei este  $m+n$  ( $m$  fiind numărul conexiunilor logice interpropoziționale parțiale diferite, inclusiv conexiunea interpropozițională totală). În cazul formulei demonstrate prin tabela din figura 2, avem:  $n=3$  (variabile propoziționale, notate  $p, q, r$ ), deci  $2^3=8$  linii. Elementele șirului finit, prin care este definit enunțul sint:  $p, q, r, q \vee r, p \wedge (q \vee r), p \wedge q, p \wedge r, (p \wedge q) \vee (p \wedge r), p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ . Deci:  $n=3, m=6$  și  $m+n=9$ .

2°. În cazul implicației (respectiv echivalenței) se folosesc și simbolii de relație „ $\Rightarrow$ ” (respectiv „ $\Leftrightarrow$ ”). Relația de implicație  $p \Rightarrow q$  semnifică faptul că „dacă  $p$  este adevărată, atunci  $q$  este adevărată”. În acest caz avem  $(p, q) \equiv (1, 1)$ . Relația  $\alpha \Rightarrow \beta$  este frecvent folosită în scrierea formalizată a teoremelor:  $\alpha \Rightarrow \beta$ , teorema directă;  $\beta \Rightarrow \alpha$ , teorema reciprocă (în cazul că teorema directă admite reciprocă adevărată);  $\neg \alpha \Rightarrow \neg \beta$ , teorema contrară directe;  $\neg \beta \Rightarrow \neg \alpha$ , teorema contrară reciprocei.

De menționat că simbolul de relație „ $\Rightarrow$ ” se folosește uneori cu un sens mai larg, anume — în toate cazurile pentru care  $\alpha \Rightarrow \beta$  este o propoziție adevărată.

Relația  $\alpha \Leftrightarrow \beta$  este frecvent folosită în înțelesul că „ $\alpha$  este adevărată dacă și numai dacă  $\beta$  este adevărată”. Această relație apare în cazul teoremelor care admit reciprocă adevărată. Este uneori înțelinită și în înțelesul dat de perechile  $(1, 1)$  și  $(0, 0)$  adică al relației „ $\equiv$ ” ( $\alpha \equiv \beta$ ). Evident, în loc de  $\alpha \equiv \beta$  se poate scrie  $\alpha \rightarrow \beta$ , pe cînd în loc de  $\alpha \leftrightarrow \beta$  nu se poate scrie totdeauna  $\alpha \equiv \beta$ .

3°. În evaluarea unei propoziții compuse, ordinea de efectuare a operațiilor logice este următoarea:  $\neg$  (nu),  $\wedge$  (și),  $\vee$  (sau),  $\rightarrow$  (implică),  $\leftrightarrow$  (echivalent).

Negația leagă deci mai tare decît conjuncția, conjuncția — mai tare decît disjuncția, conjuncția și disjuncția — mai tare decît implicația, implicația — mai tare decît echivalența.

## 21.2. Proprietăți ale operatorilor logici

În analiza și calculul logic sînt folosite o serie de proprietăți demonstrate pe baza matricelor de adevăr. În scrierea acestor proprietăți se va folosi simbolul „a” cu semnificația de propoziție totdeauna adevărată și simbolul „f” — semnificînd propoziția totdeauna falsă. Fie  $p, q, r$  trei propoziții logice. În cele ce urmează vor fi enunțate cîteva proprietăți uzuale ale operatorilor logici.

### Proprietăți ale operației „ $\wedge$ ”:

$$1^\circ. (p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$$

$$2^\circ. p \wedge q \Leftrightarrow q \wedge p$$

$$3^\circ. p \wedge a \Leftrightarrow p$$

$$4^\circ. p \wedge p \Leftrightarrow p$$

$$5^\circ. p \wedge f \Leftrightarrow f$$

— asociativitatea operației  $\wedge$

— comutativitatea operației „ $\wedge$ ”;

— „a” este elementul neutru pentru operația „ $\wedge$ ”;

— idempotența (operația  $\wedge$  este idempotentă);

— „f” este elementul absorbant pentru operația „ $\wedge$ ” (conjuncția avînd unul din factori fals este falsă)

$$6^\circ. p \wedge (q \wedge r) \Leftrightarrow (p \wedge q) \wedge (p \wedge r) \quad \text{— autodistributivitatea operației „ $\wedge$ ”}.$$

### Proprietăți ale operației „ $\vee$ ”:

$$1^\circ. (p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$$

$$2^\circ. p \vee q \Leftrightarrow q \vee p$$

$$3^\circ. p \vee f \Leftrightarrow p$$

$$4^\circ. p \vee p \Leftrightarrow p$$

$$5^\circ. p \vee a \Leftrightarrow a$$

— asociativitatea;

— comutativitatea;

— „f” este element neutru pentru operația „ $\vee$ ”;

— operația „ $\vee$ ” este idempotentă;

— „a” este elementul absorbant pentru operația „ $\vee$ ”;

$$6^\circ. p \vee (q \vee r) \Leftrightarrow (p \vee q) \vee (p \vee r) \quad \text{— autodistributivitatea operației „ $\vee$ ”}.$$

### Relații între operațiile „ $\wedge$ ”, „ $\vee$ ”, „ $\neg$ ”:

$$1^\circ. p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$$

— operația „ $\wedge$ ” este distributivă față de operația „ $\vee$ ”. Scrisă sub forma „ $(p \wedge q) \vee (p \wedge r) \equiv p \wedge (q \vee r)$ ”, această proprietate este analoagă scoaterii factorului comun.

$$2^\circ. p \wedge (q \vee r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$$

— operația „ $\vee$ ” este distributivă față de operația „ $\wedge$ ”.



**Observație:** În algebra logicii există două proprietăți de distributivitate.

3°.  $p \vee (p \wedge q) \Leftrightarrow p$ ;  $p \wedge (p \vee q) \Leftrightarrow p$  — legi de absorbție;

4°.  $(p \vee q) \wedge (p \vee \bar{q}) \Leftrightarrow p$ ;  $(p \wedge q) \vee (p \wedge \bar{q}) \Leftrightarrow p$  — legi ale excluderii;

5°. Legile lui A de Morgan:

I.  $\neg(p \wedge q) \Leftrightarrow (\neg p) \vee (\neg q)$  — negația conjuncției este logic echivalentă cu disjuncția negațiilor;

II.  $\neg(p \vee q) \Leftrightarrow (\neg p) \wedge (\neg q)$  — negația disjuncției este logic echivalentă cu conjuncția negațiilor.

Observații 1) Când nu este pericol de confuzie, semnul de negație „ $\neg$ ” se înlocuiește cu bara de supraliniere „ $\bar{\phantom{x}}$ ” și cele două legi se scriu astfel:

$$\overline{p \wedge q} \Leftrightarrow \bar{p} \vee \bar{q}; \quad \overline{p \vee q} \Leftrightarrow \bar{p} \wedge \bar{q}$$

2) Dacă într-o formulă constituită cu operatorii „ $\wedge$ ”, „ $\vee$ ”, „ $\neg$ ” se înlocuiește „ $\wedge$ ” cu „ $\vee$ ” și reciproc, se obține duala formulei date.

### Patru principii logice clasice:

1°. Principiul dublei negații: „ $\neg(\neg p) \Leftrightarrow p$ ” — dubla negație readuce propoziția la valoarea sa inițială. În afară de negația din logica formală există negația dialectică, aceasta păstrând din ceea ce s-a negat tot ceea ce este pozitiv (progresist) și dezvoltându-l pe o treaptă superioară.

2°. Principiul terțiului exclus: „ $p \vee (\neg p) \Leftrightarrow a$ ”. Cuplul disjunctiv al unei propoziții cu negația ei este o propoziție adevărată. Acest principiu este folosit în domeniul finitului, unde alternativa „ $p \vee (\neg p)$ ” se poate verifica, în fiecare caz, element cu element. De observat că chiar în domeniul finitului pot exista propoziții enunțative (respectiv afirmative) care scapă principiului terțiului exclus. De exemplu, din faptul că o propoziție „ $p$ ” nu este adevărată, nu întotdeauna rezultă că „ $\neg p$ ” este adevărată. Depășirea unor astfel de situații din gândirea logică a dus la apariția logicilor polivalente.

3°. Principiul contradicției: „ $p \wedge (\neg p) \Leftrightarrow f$ ” — o propoziție și negația ei nu pot fi simultan adevărate (sau „conjuncția în care unul din factori este negația celuilalt este falsă”). Această conexiune este o contradicție. Nerespectarea acestui principiu în cercetarea matematică constituie sursa antinomiilor (în nici un sistem de axiome nu putem avea „ $(p \Leftrightarrow a) \wedge (\neg(p \Leftrightarrow a))$ ”). Dacă într-o teorie matematică apare o astfel de situație, avem de-a face cu un paradox matematic și se impune reexaminarea sistemului de axiome al respectivei teorii.

4°. Principiul contradicției excluse. Conexiunea „ $\neg(p \wedge (\neg p))$ ” este o propoziție totdeauna adevărată. Această tautologie constituie principiul contradicției excluse „nu este adevărat că o propoziție „ $p$ ” este valabilă simultan cu negația ei”.

### Citeva legi logice și reguli uzuale raționamentului logic.

1°.  $p \rightarrow q \Leftrightarrow \bar{q} \rightarrow \bar{p}$  (legea contrapozității); la fel, „ $q \rightarrow p \Leftrightarrow \bar{p} \rightarrow \bar{q}$ ”

2°.  $(p \wedge (p \rightarrow q)) \Rightarrow q$  (legea modus ponens, sau legea detașării)

3°.  $((p \rightarrow q) \wedge (q \rightarrow r)) \Rightarrow (p \rightarrow r)$  (legea tranzitivității implicației)

4°.  $p \rightarrow q \Leftrightarrow \bar{p} \vee q$ ;  $\neg(p \rightarrow q) \Leftrightarrow p \wedge (\neg q)$  (legea implicației, respectiv legea negării implicației)

5°.  $p \rightarrow \bar{p} \Rightarrow \bar{p}$ ;  $\bar{p} \rightarrow p \Rightarrow p$ ;  $((p \rightarrow q) \wedge (p \rightarrow \bar{q})) \rightarrow \bar{p}$  (legi ale reducerii la absurd)

6°. Dintre regulile frecvent folosite în raționamentul logic se pot enumera:

I. regula detașării (a se vedea legea modus ponens);

II. regula substituției, conform căreia, înlocuind o variabilă propozițională cu una și aceeași expresie — peste tot unde variabila apare într-o expresie validă, se obține tot o expresie validă;

III. regula introducerii premizelor, conform căreia, pe parcursul unui raționament se poate introduce o premiză nouă, necesară pentru deducerea concluziei (sau pentru demonstrarea unei teoreme intermediare);

IV. regula utilizării tautologiilor.



## 21.3. Funcții logice

Din prezentarea unora dintre operatorii logici se poate observa că adevărul sau falsul unei propoziții compuse poate fi considerat ca o funcție de adevărul sau falsul propozițiilor componente.

În cazul conjuncției, al disjuncției, implicației sau al echivalenței logice, ca operatori, se constată că fiecareia din evaluările  $(f, f)$ ,  $(f, a)$ ,  $(a, f)$ ,  $(a, a)$  ale cuplului  $(p, q)$  îi corespunde prin funcția dată, conform tabelului de adevăr, o valoare și numai una din mulțimea  $\{a, f\}$ . De exemplu, pentru disjuncție, fiecareia din evaluările  $(f, f)$ ,  $(f, a)$ ,  $(a, f)$ ,  $(a, a)$  îi corespunde o valoare din  $\{f, a, a, a\}$  — în această ordine.

Deci funcțiile logice de argumente  $p, q$  sînt definite pe produsul cartezian  $\{f, a\} \times \{f, a\}$  cu valori în mulțimea  $\{a, f\}$ , conform regulilor de construcție indicate în tabele.

O funcție logică de două argumente  $p, q$  se poate scrie sub forma  $f(p, q) = p \text{ c } q$ , unde „ $p \text{ c } q$ ” înseamnă „ $p$  compus cu  $q$ ”,  $c$  fiind un conector. Fiecărei perechi de valori pentru  $(p, q)$  îi corespunde o valoare și numai una pentru „ $p \text{ c } q$ ”.

Ca și în matematică, în logică există funcții de unul sau mai multe argumente.

### Funcțiile logice de un singur argument.\*

Există patru funcții logice de un singur argument: contradicția, poziția, negația și tautologia. Ele sînt definite conform figurii 21.1. sau tabelului 21.2-a.

Contradicția (Constanta „0”)	Poziția (variabila „p”)	Negația	Tautologia (Constanta „1”)
$f_0: \{a, f\} \rightarrow \{a, f\}$ $f_0(p) = f$	$f_1: \{a, f\} \rightarrow \{a, f\}$ $f_1(p) = p$	$f_2: \{a, f\} \rightarrow \{a, f\}$ $f_2(p) = \neg p$	$f_3: \{a, f\} \rightarrow \{a, f\}$ $f_3(p) = a$

Tabela 21.2 — a

### FUNCȚII LOGICE DE UN SINGUR ARGUMENT

p	0	1	Simbol logic	Denumirea
$f_0$	0	0	0	Contradicția (sau constanta „0”)
$f_1$	0	1	p	Poziția (sau variabila „p”)
$f_2$	1	0	$\neg p$	Negația
$f_3$	1	1	1	Tautologia (sau constanta „1”)

### Funcțiile logice de două argumente\*

Funcțiile logice neechivalente de argumente  $p, q$  ale logicii bivalente, generate în actul logic al gândirii, sînt în număr de șaisprezece. Ele sînt definite, conform figurii 21.2 respectiv tabelului 21.3-a, unde s-a notat operatorul funcțional prin  $c_i$ ,  $i=0,15$ .

De asemenea, se folosește notația:

- $\alpha$  — conexiunea „ $p \text{ c } q$ ” pentru evaluarea  $(p, q) \equiv (f, f)$ ;
- $\beta$  — conexiunea „ $p \text{ c } q$ ” pentru evaluarea  $(p, q) \equiv (f, a)$ ;
- $\gamma$  — conexiunea „ $p \text{ c } q$ ” pentru evaluarea  $(p, q) \equiv (a, f)$ ;
- $\delta$  — conexiunea „ $p \text{ c } q$ ” pentru evaluarea  $(p, q) \equiv (a, a)$ .

\* „Logica modernă” de Georg Klaus, traducere, Editura Științifică și Enciclopedică, București, 1977.



Tabelul 21 3-a

p	0	0	1	1	Simbol logic	Denumirea
q	0	1	0	1		
$\begin{matrix} p & q \\ f_i \end{matrix}$	$\alpha$	$\beta$	$\gamma$	$\delta$		
$f_0$	0	0	0	0	0	Constanta „0”; funcția identic-falsă; contradicția.
$f_1$	0	0	0	1	$p \cdot q$	Produs logic „și”; conjuncția.
$f_2$	0	0	1	0	$p \vee q; p \downarrow q; \overline{p \rightarrow q}$	Negația implicației de $p, q$ ; interdicția după $q$ ; funcția excepție; exceptarea lui $q$ . Citește: „ $p$ fără $q$ ”; „ $p$ există, însă $q$ nu”
$f_3$	0	0	1	1	$p$	Variabila $p$ ; afirmație de $p$ ; afirmație de $p$ indiferent de $q$ .
$f_4$	0	1	0	0	$q \vee p; p \downarrow q; \overline{q \rightarrow p}$	Negația implicației inverse; interdicția după $p$ ; excepția inversă. Citește: „nu $p$ , ci $q$ ”, „ $q$ fără $p$ ”
$f_5$	0	1	0	1	$q$	Variabila $q$ ; afirmație de $q$ ; afirmație de $q$ indiferent $p$ .
$f_6$	0	1	1	0	$p \oplus q; p \times q; (p \wedge q) \vee (\overline{p} \wedge q)$	Disjuncția exclusivă; excluderea logică; suma modulo 2. Citește: „sau exclusiv”
$f_7$	0	1	1	1	$p \vee q$	Disjuncția logică neexclusivă; includerea; alternativă; suma logică „SAU”. Citește: „sau inclusiv”
$f_8$	1	0	0	0	$p \nmid q; \overline{p} \wedge \overline{q}$	Funcția „NICI” a lui NICOD; „NICI”; „SAU-NU”; antidisjuncția. Citește: „nici $p$ , nici $q$ ”; „ $p$ nici $q$ ”
$f_9$	1	0	0	1	$p \rightarrow q; (p \rightarrow q) \wedge (q \rightarrow p)$	Echivalența logică. Uneori se notează „ $p \equiv q$ ” și se numește „identitatea logică”; coincidența.
$f_{10}$	1	0	1	0	$\overline{q}$	Negația lui $q$ . Citește: „negație de $q$ , indiferent $p$ ”
$f_{11}$	1	0	1	1	$q \rightarrow p$	Implicația inversă; implicația de $q, p$ ; replicația (notată și „ $p \leftarrow q$ ”)
$f_{12}$	1	1	0	0	$\overline{p}$	Negația lui $p$ . Citește: „negație de $p$ , indiferent $q$ ”
$f_{13}$	1	1	0	1	$p \rightarrow q; \overline{p} \vee q$	Implicația logică; implicația materială; implicația. Citește: „dacă $p$ , atunci $q$ ” sau „ $p$ , deci $q$ ”
$f_{14}$	1	1	1	0	$p \downarrow q; \overline{p} \vee \overline{q}$	Anticonjuncția; funcția incompatibilitate” a lui SHEFFER. Citește: „ $p$ incompatibil cu $q$ ”; „fie că nu $p$ , fie că nu $q$ ”
$f_{15}$	1	1	1	1	1	Constanta „1”; identitatea logică; funcția identic-adeverată; tautologia.



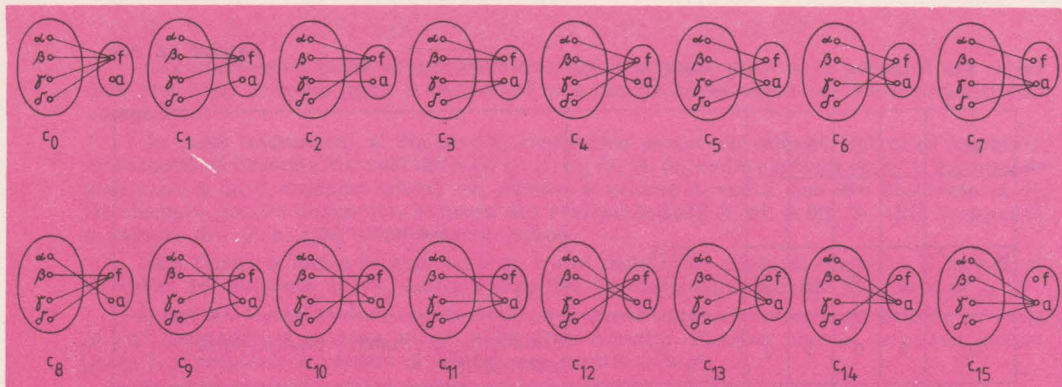


Fig. 21.2. Funcții logice de două argumente

Folosind apoi notația simbolică uzuală („o” în loc de „f” și „1” în loc de „a”), rezultă tabloul matricelor de adevăr corespunzătoare celor 16 funcții logice de argumente  $p, q$ , notate prin  $f_i$ ,  $i=0,15$ , din tabela 21.3-a.

#### Observații

1) Conform tabelii 21.3-a, funcțiile logice sînt clasificate în:

I. Legi logice (sau tautologii) „ $f_{16}$ ”,

II. Contradicții (sau antilogii) „ $f_0$ ”,

III. Funcții realizabile (sau simplu realizabile, sau propoziții contingente) „ $f_2-f_{14}$ ”.

Prin lege logică se înțelege o funcție adevărată independent de valorile luate de argumentele sale din mulțimea  $\{0, 1\}$ . Legea logică este o formulă identic adevărată, sau formulă validă, sau tautologie. Tautologiile sînt folosite în formularea unor scheme de raționament corecte, ele permițînd exprimarea unor operații logice prin alte operații logice.

Prin contradicție se înțelege o funcție logică falsă, independent de valorile luate de argumentele sale din mulțimea  $\{0, 1\}$ . De exemplu propoziția  $p \wedge \bar{p}$  este o contradicție. Contradicția este o formulă identic-falsă, sau formulă nerealizabilă.

Prin funcție realizabilă se înțelege o funcție logică adevărată în cel puțin un caz. Așa spre exemplu, propoziția „ $p \vee (q \wedge r)$ ” este o conexiune simplu-realizabilă.

2) Cele 16 linii ( $f_0-f_{15}$ ), ale tabelii 21.3-a, conțin valorile de adevăr corespunzătoare celor 16 conexiuni „ $p$  c  $q$ ” posibile. Fiecare linie, de la  $f_0$  la  $f_{15}$ , reprezintă deci o definiție a uneia din conexiunile logice de argumente  $p, q$ .

3) În tabela 21.3-a se observă că funcțiile sînt opuse simetric față de jumătatea tabelii ( $f_7$  cu  $f_8$ ;  $f_6$  cu  $f_9$ , ...,  $f_0$  cu  $f_{15}$ ).

Deci, funcțiile  $f_8, f_9, \dots, f_{15}$  pot fi definite, respectiv, ca negații ale celor opuse simetric. Acest fapt permite ameliorarea calculului logic.

4) Principalele funcții logice bivalente folosite sînt definite conform tabelilor 21.2-a și 21.3-a, astfel:

1°. Negația este funcția a cărei valoare este adevărul atunci cînd argumentul este fals și falsul atunci cînd argumentul este adevărat.

2°. Disjuncția (sau alternativa) este funcția adevărată atunci și numai atunci cînd cel puțin un argument este adevărat.

3°. Conjunția este funcția adevărată atunci și numai atunci cînd fiecare din argumentele sale sînt adevărate.

4°. Implicația este funcția care este falsă atunci și numai atunci cînd antecedentul (ipoteza „ $p$ ”) este adevărată, iar consecventul (concluzia „ $q$ ”) este fals.

5°. Echivalența este funcția care este adevărată atunci și numai atunci cînd argumentele sale sînt în același timp adevărate sau false.

\* „Logica simbolică”, de Gheorghe Enescu, Editura Științifică, București, 1971.



6°. Excluderea este funcția adevărată atunci și numai atunci când numai un argument are ca valoare de adevăr adevărul.

7°. Antidisjunția (sau funcția „NICI”) este funcția adevărată dacă și numai dacă toate argumentele sale sînt false.

8°. Anticonjunția (sau funcția „INCOMPATIBILITATE”) este funcția falsă dacă și numai dacă toate argumentele sale sînt adevărate.

5) Numărul total al funcțiilor de  $n$  argumente ale logicii bivalente este dat de formula  $N=2^{2^n}$  (numărul funcțiilor de două argumente fiind  $N=2^{2^2}=16$ , iar cel al funcțiilor de un singur argument,  $N=2^{2^1}=4$ ).

În general, se demonstrează că numărul operatorilor logici de  $n$  argumente ce se pot construi într-o logică cu  $m$  valori de adevăr este dat de formula  $N=m^{m^n}$ .

6) Considerînd ca primitiv grupul de operatori „ $\wedge$ ,  $\vee$ ,  $\neg$ ”, al algebrei Boole, cu ajutorul acestora se pot defini toți ceilalți. De altfel, toți operatorii logicii propozițiilor se pot exprima printr-un singur operator „antidisjunția”. Exprimarea operatorilor prin antidisjunție este greoaie. Totuși circuitul care modelează fizic antidisjunția este foarte simplu.

7) Logici polivalente. Logicile în care se postulează existența a trei sau mai multe valori distincte pentru caracterizarea valorii de adevăr a unei propoziții sînt numite logici polivalente. De exemplu: logica trivalentă admite ca valori de adevăr-adevărul, falsul, posibilul (sau îndoielnicul); logica tetravalentă admite patru valori etc.

În logica trivalentă, spre exemplu, din faptul că o propoziție „ $p$ ” nu este adevărată nu rezultă adevărul propoziției „ $\neg p$ ” — așa cum se știe din logica bivalentă.

Logicile polivalente au largi implicații în algebra modernă, în teoria rețelelor electrice cu relee și contacte, în fizică etc.

## 21.4. Problema deciziei

Există mai multe metode pentru a decide dacă o expresie logică reprezintă o tautologie, o contradicție, sau o expresie simplu — realizabilă.

I. metoda matriceală;

II. metoda eliminării treptate a necunoscutelor — utilizînd raționamentul logic bazat pe reguli ce decurg din definițiile operațiilor logice și proprietățile acestora;

III. metoda formelor normale.

În logica propozițiilor, spunem că o expresie logică are forma disjunctivă dacă operatorul său principal este „ $\vee$ ”, iar termenii săi sînt conjuncții. De asemenea, o expresie logică are forma conjunctivă dacă operatorul său este „ $\wedge$ ”, iar factorii sînt disjuncții.

În calculul logic, orice expresie logică poate fi adusă la o așa-zisă formă normală (poate fi normalizată). Normalizarea unei expresii logice înseamnă transformarea acesteia, pe baza legilor logicii, în expresii simple echivalente cu ea. În algebra logicii, formele normale sînt construite cu ajutorul operatorilor de bază ( $\wedge$ ,  $\vee$ ,  $\neg$ ).

A aduce o expresie logico-propozițională la o formă normală înseamnă a o aduce la acea formă în care să nu intervină decît operatorii de bază, negația căzînd numai pe simbolurile pentru propozițiile elementare.

O expresie logică este adusă la forma normală dacă este fie o disjuncție de conjuncții, fie o conjuncție de disjuncții.

Orice formă normală are deci un operator principal. După numele operatorului principal, care dă și denumirea funcției, avem două forme normale: forma normală conjunctivă (care este o conjuncție de disjuncții) și forma normală disjunctivă (care este o disjuncție de conjuncții).

O expresie se aduce la una din cele două forme pentru a decide mai ușor dacă este o tautologie, o contradicție, sau o funcție realizabilă.

Operatorul care dă denumirea formei normale nu apare în membrii expresiei. De exemplu, expresia  $p \wedge (q \vee r)$  nu este o formă normală, pe cînd expresia  $(p \vee \bar{p} \vee \bar{q}) \wedge (q \vee \bar{q} \vee \bar{p})$  este o formă normală conjunctivă, iar  $(p \wedge \bar{p} \wedge \bar{q}) \vee (p \wedge q \wedge \bar{q})$ , este o formă normală-disjunctivă.

În formele normale sînt folosiți termenii:

1°. produs elementar (o conjuncție de propoziții simple și de negații ale lor);

2°. sumă elementară (o disjuncție de propoziții simple și de negații ale lor);

## 21. OPERATORI LOGICI



3°. formă normală disjunctivă (sumă de produse elementare);

4°. formă normală conjunctivă (produs de sume elementare).

În calculul formelor normale sînt folosite proprietățile operatorilor logici:  $p \wedge p \equiv p$ ,  $p \vee p \equiv p$ ,  $p \wedge f \equiv f$ ,  $p \vee f \equiv p$ ,  $p \wedge a \equiv p$ ,  $p \vee a \equiv a$ ,  $p \wedge \bar{p} \equiv f$ ,  $p \vee \bar{p} \equiv a$ .

Pe baza proprietăților sînt date o serie de reguli pentru validarea formelor normale.

1°. În actul deciziei se ține seama de legile logice  $\alpha \vee \bar{\alpha} \equiv a$  (tautologie) și  $\alpha \wedge \bar{\alpha} \equiv f$  (contradicție):

I. dacă în fiecare membru al formei normale conjunctive se află o expresie de forma  $\alpha \vee \bar{\alpha}$ , atunci forma normală este o tautologie;

II. dacă în fiecare membru al formei normale conjunctive se află o expresie de forma  $\alpha \wedge \bar{\alpha}$ , atunci forma normală este o contradicție;

III. dacă o expresie adusă la forma normală nu este nici tautologie, nici contradicție, ea este o funcție realizabilă.

2° În calculul pentru obținerea deciziei se efectuează transformări asupra expresiei logice în vederea eliminării acelor operatori care nu aparțin mulțimii operatorilor de bază.

De exemplu, se înlocuiește  $\alpha \rightarrow \beta$  prin  $\bar{\alpha} \vee \beta$ ,  $\alpha \downarrow \beta$  prin  $\overline{\alpha \vee \beta}$  etc. De asemenea, se elimină negațiile care nu cad pe variabile, ele trecînd pe variabile prin folosirea indeosebi a legilor lui A de Morgan:  $\overline{\alpha \wedge \beta} \equiv \bar{\alpha} \vee \bar{\beta}$  și  $\overline{\alpha \vee \beta} \equiv \bar{\alpha} \wedge \bar{\beta}$ . Se aduce apoi expresia astfel obținută la forma normală dorită — pe baza legilor de distributivitate, asociativitate și comutativitate. Se ierarhizează în final operatorii, stabilindu-se cel principal și cel secundar.

*Aplicație\**). Să se decidă dacă expresia logică „ $((p \rightarrow \bar{q}) \wedge q) \rightarrow \bar{p}$ ” este o lege logică.

Soluție. Se aduce expresia la forma normală conjunctivă și se decide.

a) În prima etapă se elimină functorii care nu apar în forma normală conjunctivă (se elimină deci implicația):

$$((p \rightarrow \bar{q}) \wedge q) \rightarrow \bar{p} \equiv (\overline{(p \rightarrow \bar{q}) \wedge q}) \vee \bar{p} \equiv (\overline{(\bar{p} \vee \bar{q}) \wedge q}) \vee \bar{p} \equiv \dots$$

b) În cea de-a doua etapă se elimină negația care cade pe propoziția compusă și avem în continuare

$$\equiv (\overline{(\bar{p} \vee \bar{q}) \vee q}) \vee \bar{p} \equiv (\overline{(\bar{p} \wedge \bar{q}) \vee q}) \vee \bar{p} \equiv ((p \wedge q) \vee \bar{q}) \vee \bar{p} \equiv \dots$$

c) În cea de-a treia etapă se ierarhizează functorii (conectorii), astfel încît să avem o conjuncție de disjuncții (conjuncția — functor principal, iar disjuncția — functor secundar). Se aplică asociativitatea și apoi distributivitatea disjuncției față de conjuncție:

$$\equiv (p \wedge q) \vee (\bar{q} \vee \bar{p}) \equiv (p \vee (\bar{q} \vee \bar{p})) \wedge (q \vee (\bar{q} \vee \bar{p})) \equiv \dots$$

d) În cea de-a patra etapă se aplică asociativitatea disjuncției și avem:

$$\equiv ((p \vee \bar{q} \vee \bar{p}) \wedge (q \vee \bar{q} \vee \bar{p})) \equiv (p \vee \bar{p} \vee \bar{q}) \wedge (q \vee \bar{q} \vee \bar{p}). \text{ — deci, forma normală conjunctivă) } \equiv \dots$$

e) Toți membrii fiind valizi, în virtutea principiului terțiului exclus avem:

$$\equiv (a \vee \bar{q}) \wedge (a \vee \bar{p}) \equiv a \wedge a \equiv a.$$



## 22. 1. Scurtă prezentare

Algebra Boole este o mulțime nevidă  $M$  înzestrată cu două operații interne binare, denumite sumă booleană și produs boolean și o operație internă unară de complementare, care se bucură de anumite proprietăți. Această algebră folosește un sistem de simboluri și reguli cu ajutorul cărora se pot face operații booleene, analoage celor din algebră. Enumerăm astfel:

- i) Simbolurile pentru elementele mulțimii  $M$ ,  $(a, b, c, \dots; x, y, z, \dots; p, q, r, \dots)$ .
- ii) Elementul nul — simbolizat prin  $0$  și elementul universal — simbolizat prin  $1$  sau  $u$ .
- iii) Operatorii binari  $+$  și  $\cdot$  și operatorul unar de complementare — simbolizat prin supralinierea elementului, de exemplu „ $\bar{a}$ ”.

Simbolurile și operatorii algebrei Boole pot fi interpretați în mod diferit, depinzând de mulțimea sau de clasa de elemente pe care sînt definiți. Astfel, operația de adunare booleană, notată cu  $+$ , corespunde fie reuniunii „ $U$ ” (definită pe mulțimea părților unei mulțimi), fie disjuncției „ $V$ ” (definită pe o clasă de propoziții bivalente) etc.

Produsul boolean corespunde fie intersecției „ $\cap$ ” (pe mulțimea părților unei mulțimi), fie conjuncției propozițiilor „ $\wedge$ ” etc. Operația notată prin supraliniere corespunde complementării mulțimilor, respectiv negației propozițiilor.

- iv) Semnele pentru funcții, pentru egalitate (sau identitate), parantezele „( )” etc.

**Definiție.** Se numește algebră booleană o mulțime nevidă  $M$  înzestrată cu două operații binare „ $+$ ” și „ $\cdot$ ” și cu o aplicație „non” a mulțimii considerate în ea însăși, astfel încît pentru orice  $a, b, c \in M$  sînt satisfăcute proprietățile:

1.  $a + b = b + a$                        $a \cdot b = b \cdot a$                       (comutativitatea);
2.  $(a + b) + c = a + (b + c)$  ,     $(a \cdot b) \cdot c = a \cdot (b \cdot c)$                       (asociativitatea);
3.  $a + a = a$                               ,     $a \cdot a = a$                               (idempotența);
4.  $a + (a \cdot b) = a$                       ,     $a \cdot (a + b) = a$                       (absorbția);
5.  $a + b \cdot c = (a + b) \cdot (a + c)$  ,     $a \cdot (b + c) = a \cdot b + a \cdot c$  (distributivitatea);

---

\* Denumirea de algebră booleană vine de la numele matematicianului și logicianului George Boole (1815—1865).

Noțiunile selectate aici, din bibliografia citată, pot folosi ca teme pentru cercurile de matematică și informatică. Menționăm ca lucrări de bază, din care s-au folosit exemple.

1. Logica matematică și algebra booleană”, de Ioana Bărbat și Boldur Bărbat, E.D.P. București, 1978.
2. „Matematica aplicată în tehnica de calcul”, de Ioan Tomescu și Adrian Leu, E.D.P. București, 1978.
3. „Informatica”, de M. Lovin, P. Preoteasa, C. Popovici, E.D.P., Buc., 1972.



6. Există un element unic în  $M$ , notat cu simbolul „0”, numit „prim element”, sau „element nul”, cu proprietățile:

$$a+0=a, \quad a \cdot 0=0;$$

7. Există un element unic în  $M$ , notat cu simbolul „1”, sau cu „u”, numit „ultim element”, sau „element universal”. cu proprietățile:

$$a+1=1, \quad a \cdot 1=a.$$

8. Legea dublei negații (involuția):

$$\bar{\bar{a}}=a.$$

9. Principiul contradicției:

$$a \cdot \bar{a}=0.$$

10. Principiul terțiului exclus:

$$a+\bar{a}=1.$$

11. Legile lui De Morgan:

$$\overline{a+b}=\bar{a} \cdot \bar{b} \quad ; \quad \overline{a \cdot b}=\bar{a}+\bar{b}.$$

Propozițiile enumerate în definiția algebrei boolene nu constituie un sistem de axiome independent. De exemplu, luând absorbția ca axiomă, se poate demonstra idempotența; sau, luând ca axiome principiul terțiului exclus, precum și existența unui prim element și a unui ultim element, se pot demonstra legile lui De Morgan și legea dublei negații. Ținând seama de acest fapt, putem restringe numărul axiomelor ce definesc o algebră booleană la următoarele:

1.  $a+b=b+a$ ;  $a \cdot b=b \cdot a$  (proprietatea de comutativitate);
2.  $a+(b \cdot c)=(a+b)(a+c)$ ;  $a \cdot (b+c)=a \cdot b+a \cdot c$  (proprietatea de distributivitate);
3.  $a+0=a$ ;  $a \cdot 1=a$  (existența unui prim și a unui ultim element);
4.  $a+\bar{a}=1$ ;  $a \cdot \bar{a}=0$  (proprietăți ale complementării).

În algebra booleană operatorii „+” și „ $\cdot$ ” nu au semnificația celor din aritmetica elementară, aceste semne putând fi interpretate în mod diferit.

Pentru acest motiv algebra booleană are mai multe modele particulare, pe care le redăm în tabloul din figura 22.1. Aci este vorba de mulțimi sau de clase de elemente, înzestrate cu operațiile denumite: adunarea booleană, produsul boolean și complementarea booleană, care satisfac axiomele algebrei Boole. Astfel, avem ca modele particulare ale algebrei Boole: logica propozițiilor bivalente, algebra mulțimii părților unei mulțimi, algebra circuitelor cu contacte și relee, algebra booleană binară și algebra vectorilor booleni (fig. 22.1).

În cadrul acestor modele particulare ne vom opri, pe scurt, asupra algebrei boolene formată din două elemente (algebra  $L_2$ ). Conform proprietăților 6) și 7), din definiția Algebrei Boole, rezultă că o algebră booleană are cel puțin două elemente. Algebra  $L_2$  are ca elemente chiar elementele neutre ale celor două operații interne ale Algebrei Boole: „0” și „1”. Avem deci mulțimea  $L_2=\{0,1\}$ , pe care se definesc cele trei operații interne astfel:







Algebra booleană	Elemente	Operații			Prim element	Ultim element
	a, b	+ adunare	• produs	"-" ( $\bar{a}$ ) sau apostrof ( $a'$ )	0	1
Logica propozițiilor	p, q	$\vee$	$\wedge$	$\neg, -, \sim$	clasa contradicțiilor $\emptyset$	clasa tautologiilor $\mathcal{E}$
Teoria mulțimilor	A, B	$\cup$	$\cap$	$\complement_M$	$\Phi$ (mulțimea vidă)	M (mulțimea de referință)
Algebra schemelor cu contacte și rele					(conexiunea care nu este niciodată permeabilă curentului electric)	(conexiunea permanent permeabilă curentului electric)
Algebra $L_2$	0, 1	+	•	-	0	1
Algebra vectorilor cu n componente binare	v, w	+	•	"-"	$0 = (0, 0, \dots, 0)$ (vector cu toate n componentele nule)	$1 = (1, 1, \dots, 1)$ (vector cu toate n componentele egale cu 1)

Fig. 22.1. Modele particulare ale algebrei booleene

$$0+0=0,$$

$$0 \cdot 0=0,$$

$$\bar{0}=1$$

$$0+1=1,$$

$$0 \cdot 1=0,$$

$$\bar{1}=0.$$

$$1+0=1,$$

$$1 \cdot 0=0,$$

$$1+1=1,$$

$$1 \cdot 1=1,$$

Se observă că adunarea booleană nu coincide cu adunarea din aritmetică. În algebra  $L_2$ , adunarea este definită astfel:

$$a+b=\max(a, b), \forall a, b \in L_2$$

Înmulțirea booleană coincide cu înmulțirea din aritmetică pe mulțimea  $\{0, 1\}$ :

$$a \cdot b=\min(a, b), \forall a, b \in L_2$$

Complementarea este interpretată algebric de relația

$$\bar{a}=1-a, \forall a \in L_2$$



Proprietățile enumerate în definiția algebrei Boole se verifică în algebra  $L_2$  înlocuind elementele  $a, b, c$  prin 0 și 1 (în toate combinațiile posibile). Dacă se înlocuiește „0” cu „f” și „1” cu „a”, atunci se regăsesc tabelele de adevăr din logica bivalentă a propozițiilor, unde: „+” reprezintă alternativa „V”, „.” reprezintă conjuncția „^”, iar supralinierea „—” reprezintă negația „¬”. Elementele algebrei  $L_2$  pot deci să descrie și valoarea logică a unor propoziții. Cu ajutorul tabelelor de adevăr vor fi verificate proprietățile absorbției ( $a + ab = a$ ) și distributivității operației „+” față de operația „.” ( $a + b \cdot c = (a + b) \cdot (a + c)$ ) — tabelele 22.1-a și 22.1-b.

Tabelul 22.1-a

a	b	a+b	a·(a+b)	ab	a+ab	a(a+b)=a	a+ab=a
0	0	0	0	0	0	1	1
0	1	1	0	0	0	1	1
1	0	1	1	0	1	1	1
1	1	1	1	1	1	1	1

Tabelul 22.1 — b

a	b	c	b·c	a+b·c	a+b	a+c	(a+b)·a+c	a+b·c=(a+b)·(a+c)
0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	0	1
0	1	0	0	0	1	0	0	1
0	1	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1	1
1	0	1	0	1	1	1	1	1
1	1	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1

Elementele „0” și „1” ale algebrei  $L_2$  pot fi interpretate ca reprezentând cei doi dipoli elementari din algebra circuitelor (figura 22.2-a), iar operațiile „·” și „+” reprezentând legarea dipolilor\* în serie, respectiv derivație (figurile 22.2 (b,c)).

Doi dipoli sînt considerați echivalenți dacă amîndoi fie permit trecerea curentului electric, fie nu o permit. Echivalența se notează prin semnul „≡” sau prin semnul „=”. Operația de complementare corespunde schimbării stării dipolului.

\* Prin dipol înțelegem o rețea cu două borne



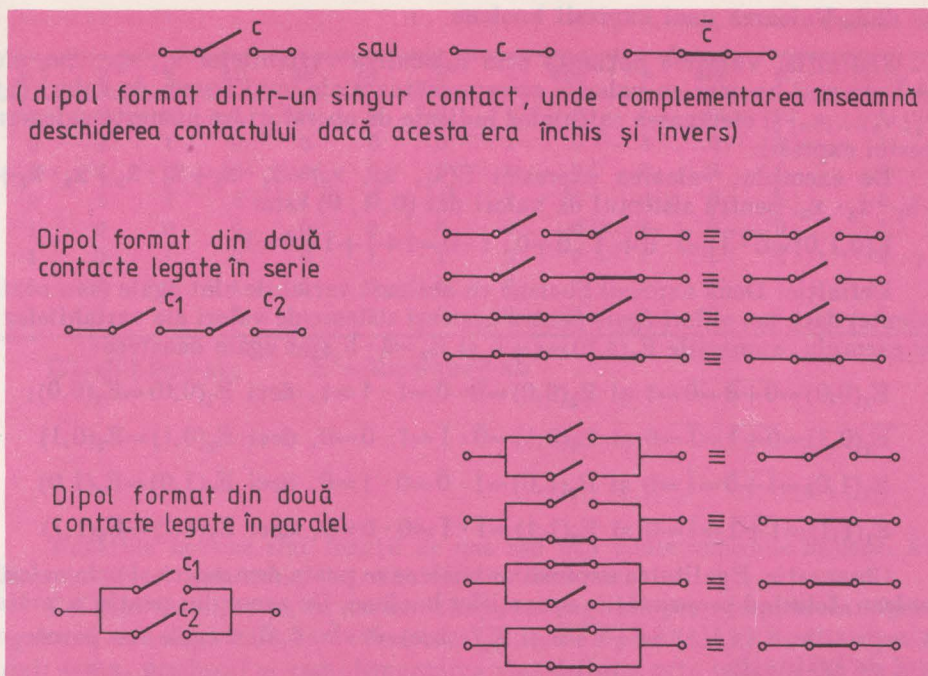


Fig. 22.2. a) — Dipol format dintr-un singur contact; b) — Legarea în serie a doi dipoli c) — Legarea în paralel a doi dipoli

## 22.2. Expresii boolene

Ca și în cazul aritmeticii sau al algebrei, într-o algebră booleană se pot construi formule sau expresii boolene prin legarea între ele a elementelor mulțimii cu ajutorul operatorilor booleni și al parantezelor (rotunde). Spre exemplu, formula „ $ac(b+\bar{b})+abc\bar{a}$ ” este o expresie booleană; ea conține nedeterminate (litere ale alfabetului latin), conectori și paranteze.

**2.1. Definiție.** Se numește expresie booleană orice expresie rezultată prin aplicarea de un număr finit de ori a operatorilor  $+$ ,  $.$ ,  $-$  unor elemente determinate sau nedeterminate ale unei algebre boolene.

**Observații.** 1. Expresiile: „ $a+.b$ ” și „ $x(\wedge y)$ ” nu sînt expresii boolene.

2. Ordinea în care se efectuează operațiile cu expresii boolene este cea indicată de paranteze sau de prioritatea operatorilor. Complementarea are cea mai mare prioritate, fiind urmată (în ordine) de operația „ $.$ ”, operația „ $+$ ” fiind cea mai puțin prioritară.

3. În orice algebră booleană se pot construi expresii boolene folosind notațiile proprii respectivei algebre.



## 2.2. Valoarea unei expresii boolene

**Definiție.** Valoarea obținută prin înlocuirea variabilelor  $x_1, x_2, \dots, x_n$  ale unei expresii boolene cu valorile corespunzătoare ale unui sistem de valori dat  $(v_1, v_2, \dots, v_n)$  și efectuarea calculelor indicate de operatori, se numește valoarea acestei expresii.

De exemplu, valoarea expresiei  $E(x_1, x_2, x_3) = \bar{x}_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot \bar{x}_3 + x_2 \cdot \bar{x}_3 + x_1 \cdot x_2 \cdot x_3$ , pentru sistemul de valori dat  $(0, 1, 0)$  este:

$$E(0,1,0) = \overline{0 \cdot 1} + \overline{0 \cdot 0} + 1 \cdot \overline{0 \cdot 0} = 1 + 1 + 1 + 0 = 1$$

**Definiție.** Două expresii boolene cu aceleași variabile sînt egale (sau echivalente) dacă iau valori egale pentru aceleași sisteme de valori ale variabilelor. De exemplu, expresiile  $E_1(a,b) = \overline{a+b}$  și  $E_2 = \bar{a} \cdot \bar{b}$  sînt egale deoarece:

$$E_1(0,0) = \overline{0+0} = \bar{0} = 1 \text{ și } E_2(0,0) = \bar{0} \cdot \bar{0} = 1 \cdot 1 = 1, \text{ deci } E_1(0,0) = E_2(0,0);$$

$$E_1(0,1) = \overline{0+1} = \bar{1} = 0 \text{ și } E_2(0,1) = \bar{0} \cdot \bar{1} = 1 \cdot 0 = 0, \text{ deci } E_1(0,1) = E_2(0,1)$$

$$E_1(1,0) = \overline{1+0} = \bar{1} = 0 \text{ și } E_2(1,0) = \bar{1} \cdot \bar{0} = 0 \cdot 1 = 0, \text{ deci } E_1(1,0) = E_2(1,0)$$

$$E_1(1,1) = \overline{1+1} = \bar{1} = 0 \text{ și } E_2(1,1) = \bar{1} \cdot \bar{1} = 0 \cdot 0 = 0, \text{ deci } E_1(1,1) = E_2(1,1)$$

**Observație.** Egalitatea expresiilor boolene se poate demonstra și prin calcul boolean, folosind proprietățile operațiilor boolene. De exemplu, pentru a arăta că expresiile  $E_1(a,b,c) = \overline{a+b+c}$  și  $E_2(a,b,c) = \bar{a} \cdot \bar{b} \cdot \bar{c}$  sînt egale, se parcurge șirul de egalități:

$$\overline{a+b+c} = \overline{(a+b)+c} = \overline{(a+b)} \cdot \bar{c} = (\bar{a} \cdot \bar{b}) \cdot \bar{c} = \bar{a} \cdot \bar{b} \cdot \bar{c}$$

De asemenea, egalitatea expresiilor boolene mai poate fi demonstrată fie prin diagrame Euler-Venn, fie prin tabele de adevăr.

În acest mod vor fi demonstrate egalitățile:

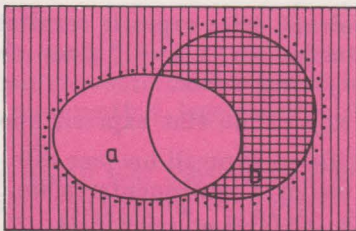
i)  $a + (\bar{a} \cdot b) = a + b$  (prin calcul și prin diagrame Euler-Venn, fig. 22.3 a);

ii)  $a \cdot b + b \cdot c + a \cdot c = \bar{a} \cdot b \cdot c + a \cdot \bar{b} \cdot c + a \cdot b$  (prin tabela de adevăr 22.2.)

Calculul boolean:

$$a + (\bar{a} \cdot b) = (a + \bar{a}) \cdot (a + b) = 1 \cdot (a + b) = a + b$$

Diagrama Euler-Venn este dată în figura 22.3-a, iar tabela de adevăr în tabela 22.2.



$\bar{a}$  - hașur vertical

$\bar{a} \cdot b$  - hașur orizontal

$a + \bar{a} \cdot b$  - reprezentată punctat  
(este suma  $a+b$ )

Fig. 22.3. — a  $a + (\bar{a} \cdot b) = a + b$



a	b	c	ab	bc	ca	E	$\bar{a}$	$\bar{a}bc$	$\bar{b}$	$a\bar{b}c$	E'	$E=E'$
0	0	0	0	0	0	0	1	0	1	0	0	1
0	0	1	0	0	0	0	1	0	1	0	0	1
0	1	0	0	0	0	0	1	0	0	0	0	1
0	1	1	0	1	0	1	1	1	0	0	1	1
1	0	0	0	0	0	0	0	0	1	0	0	1
1	0	1	0	0	1	1	0	0	1	1	1	1
1	1	0	1	0	0	1	0	0	0	0	1	1
1	1	1	1	1	1	1	0	0	0	0	1	1

$$E = ab + bc + ac; \quad E' = \bar{a}bc + a\bar{b}c + ab$$

### 22.3. Funcții boolene

Funcțiile boolene sînt funcții de una sau mai multe variabile boolene. Atît domeniul cit și codomeniul lor sînt algebre boolene.

Cel mai solicitat și, în același timp, cel mai simplu exemplu de algebră booleană este algebra  $B_2$  (notată uneori  $L_2$ ), unde  $B_2 = \{0, 1\}$ , iar operațiile algebrei: suma, produsul și complementarea sînt definite prin tabelele 22.3(a,b,c).

Tabelul 22.3 — a

	x	
	0	1
y	0	0
	1	0

Tabelul 22.3 — b

	x	
	0	1
y	0	0
	1	1

Tabelul 22.3 — c

	x	0	1
$\bar{x}$	1	0	

O funcție booleană atașează unui sistem (unei combinații) de valori atribuite variabilelor boolene un element unic al codomeniului, numit valoarea funcției boolene pentru sistemul de valori considerat.

Prin definiție, se numește funcție booleană (sau funcție binară) de  $n$  variabile o funcție definită pe produsul cartezian  $B_2 \times B_2 \times \dots \times B_2$  cu valori în  $B_2$ .

Valoarea funcției pentru un sistem dat  $(v_1, v_2, \dots, v_n)$  de valori 0 și 1 se notează  $f(v_1, v_2, \dots, v_n)$ . Se observă de aci că domeniul de definiție al funcției fiind un produs cartezian, un element al acestuia este un sistem ordonat alcătuit din  $n$  elemente de „0” sau „1” și fiecărui astfel de sistem îi corespunde valoarea „0” sau „1”.

Această corespondență poate fi dată:

- fie printr-un tabel (vezi tabelele 22.4-a, 22.4-b, 22.4-c);
- fie prezentîndu-i valorile pentru fiecare din perechile produsului cartezian  $B_2 \times B_2 = \{(0,0), (0,1), (1,0), (1,1)\}$ , spre exemplu:  $f(0,0)=1, f(0,1)=1, f(1,0)=0, f(1,1)=1$ ;
- fie reprezentînd corespondența printr-o diagramă (fig. 22.4).



Tabelul 22.4 — a

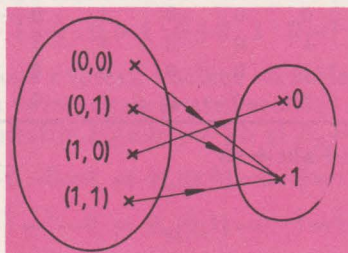
x	y	f
0	0	1
0	1	1
1	0	0
1	1	1

Tabelul 22.4 — b

xy	0	0	0	1	1	0	1	1
f		1		1		0		1

Tabelul 22.4 — c

	y	0	1
x	0	1	1
1	0	1	



Tabelul 22.4 — d

p	q	$p \rightarrow q$
$\varphi$	$\varphi$	a
$\varphi$	a	a
a	$\varphi$	$\varphi$
a	a	a

Fig. 22.4. Diagrama implicației

Dacă în tabela 22.4-a, sau în diagrama din figura 22.4, se înlocuiește o prin  $\varphi$  (fals) și 1 prin a (adevăr), atunci se obține tabela de definiție a implicației propozițiilor deci, tabela 22.4-d. Deci, funcția f din tabela 22.4-a atașează propozițiilor p, q formula  $p \rightarrow q$ . Cu alte cuvinte, formula  $p \rightarrow q$  din calculul propozițional este o funcție booleană — algebra propozițiilor fiind o algebră booleană. În tabela 22.5-a este definită funcția

$$f = x_1x_2 + x_2x_3 + x_1x_3.$$

Pe primele trei coloane sînt așezate cele opt sisteme de cîte trei valori ale nedeterminantelor  $x_1, x_2, x_3$  — în ordinea scrierii în baza 2 a numerelor 0, 1 2, 3, 4, 5, 6, 7. În tabela 22.5-b se evidențiază faptul că o funcție booleană poate fi generată de una sau mai multe expresii boolene, expresiile care generează aceeași funcție fiind egale.

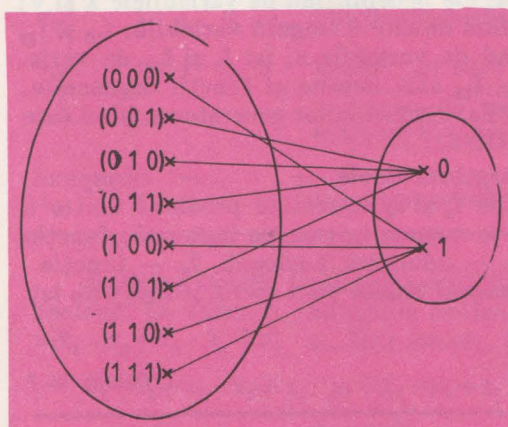
De exemplu, expresiile:  $\epsilon_1 = x_1x_2x_3 + x_1x_2\bar{x}_3 + \bar{x}_2\bar{x}_3$ ,  $\epsilon_2 = x_1x_2 + \bar{x}_2\bar{x}_3$  și  $\epsilon_3 = (x_1 + \bar{x}_2) \cdot (x_2 + \bar{x}_3)$  generează aceeași funcție  $f: B_2^3 \rightarrow B_2$  (dată prin diagrama din figura 22.5).

Tabelul 22.5 — a

$x_1$	$x_2$	$x_3$	$f = x_1x_2 + x_2x_3 + x_1x_3$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



$x_1$	$x_2$	$x_3$	$\bar{x}_2$	$\bar{x}_3$	$x_1x_2x_3$	$x_1x_2\bar{x}_3$	$\bar{x}_2 \cdot \bar{x}_3$	$\varepsilon_1$	$x_1x_2$	$\varepsilon_2$	$x_1 + \bar{x}_2$	$\varepsilon_3$	$\varepsilon_4$
0	0	0	1	1	0	0	1	1	0	1	1	1	1
0	0	1	1	0	0	0	0	0	0	0	1	0	0
0	1	0	0	1	0	0	0	0	0	0	0	1	0
0	1	1	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	1	0	0	1	1	0	1	1	1	1
1	0	1	1	0	0	0	0	0	0	0	1	0	0
1	1	0	0	1	0	1	0	1	1	1	1	1	1
1	1	1	0	0	1	0	0	1	1	1	1	1	1

Fig. 22.5.  $f: B_3 \rightarrow B_2$ 

Așa cum rezultă din definiția funcției boolene, domeniul său de definiție este o mulțime finită  $B_2 \times B_2 \times \dots \times B_2$ , notată  $B_2^n$ , iar codomeniul este  $B_2 = \{0, 1\}$ . Există deci un număr finit de funcții boolene de  $n$  variabile,  $f: B_2^n \rightarrow B_2$ , numărul lor „ $N$ ” fiind dat de formula

$$N = 2^{2^n}.$$

Mulțimea acestor funcții, înzestrată cu operațiile „+”, „·”, „—”, formează o algebră Boole. Elementul nul al acestei algebre este funcția identic nulă, iar elementul universal — funcția identic 1. Aceste funcții sînt constante.

Cele  $2^{2^n}$  funcții boolene se numerează cu  $f$  indexat „ $f_n$ ”, indexul  $n$  fiind număr natural,  $n \in \{0, 2^{2^n} - 1\}$ . O funcție booleană corespunde unui șir de  $2^n$  valori de „0” sau „1”. De exemplu, funcția prezentată prin tabela 22.4-a are codomeniul format din  $2^2 = 4$  valori de „0” sau „1” și anume mulțimea  $\{1, 1, 0, 1\}$ . Aceste valori formează un număr binar „1101”, adică numărul 13 în baza zece ( $1101_2 = 13_{10}$ ). Numărul astfel obținut este indexul funcției „ $f_{13}$ ” dată prin tabela definițiilor din figura 22.7.

Dacă în formula  $N = 2^{2^n}$  punem  $n = 1$ , obținem  $N = 4$  — deci patru funcții boolene de o variabilă. Ele sînt prezentate prin tabela 22.6.



Tabelul 22.6

Funcția	$f(x)=0$	$f(x)=x$	$f(x)=\bar{x}$	$f(x)=1$
Va- lorile varia- bilei $x$	Funcția identic nulă, sau „cons- tantă 0”	Funcția identică, sau „variabila $x$ ”	Funcția com- plement, sau „funcția negație”	Funcția identic 1, sau „constan- ta 1”
0	0	0	1	1
1	0	1	0	1

Dacă în formula  $N=2^{2^n}$  punem  $n=2$ , obținem cele 16 funcții boolene de două variabile  $x$  și  $y$ , prezentate în tabela de definiție 22.7, cu denumirile și expresiile care le generează.

Observații. 1. Cele 16 funcții boolene de două variabile au fost tabelate (în 22.7) în ordinea crescătoare a numerelor în baza 2.

2. Funcțiile  $f_0$  și  $f_{15}$  sînt constante, ele nedepinzînd de variabilele  $x$  și  $y$ ;  $f_3$  și  $f_5$  sînt funcții identice, ele depinzînd de cîte o singură variabilă;  $f_{10}$  și  $f_{12}$  sînt funcții complement;  $f_3$  și  $f_{12}$  depind de variabila  $x$ , iar  $f_5$  și  $f_{10}$  de variabila  $y$ . Cele șase funcții:  $f_0, f_{15}, f_3, f_5, f_{10}, f_{12}$  sînt numite și funcții degenerare. Celelalte funcții depind de două variabile, expresiile lor generatoare fiind date în tabelă și sub forma canonică disjunctivă.

3. Cele trei operații de bază ale algebrei Boole  $+$ ,  $\cdot$ ,  $-$  corespund, respectiv, funcțiilor  $f_1$ ,  $f_7$  și  $f_{12}$ . Funcțiile  $f_2$  și  $f_4$  reprezintă produsul dintre o variabilă și complementul celeilalte și se numesc intersecție indirectă; funcția  $f_6$  este numită și sumă disjunctivă,  $f_7$  — adunarea booleană,  $f_8$  — funcția „NICI”, ea luînd valoarea 1 atunci și numai atunci cînd nici  $x$  și nici  $y$  nu iau valoarea 1.

Tabelul 22.7

$f_n$	$xy$	00	01	10	11	Denumirea funcției	Forma canonică standar- dizată a funcției
$f_0$		0	0	0	0	Funcția constantă „0” (constanta „0”)	$f_0=0$
$f_1$		0	0	0	1	Funcția intersecție (conjuncție, produs logic); funcția „ȘI”	$f_1=x \cdot y$
$f_2$		0	0	1	0	Funcția intersecție indirectă (interdicția după $y$ )	$f_2=x \cdot \bar{y}$ (sau $f_2=x \nabla y$ )
$f_3$		0	0	1	1	Funcția identic $x$ (variabila $x$ )	$f_3=x$
$f_4$		0	1	0	0	Funcția intersecție indirectă, sau inversă (interdicția după $x$ )	$f_4=\bar{x} \cdot y$ (sau $f_4=y \nabla x$ )
$f_5$		0	1	0	1	Funcția identic $y$ (variabila $y$ )	$f_5=y$
$f_6$		0	1	1	0	Funcția sumă disjunctivă — suma modulo 2 („sau exclusiv”, diferența simetrică, antivalența)	$f_6=x \oplus y$ ; $f_6=\bar{x} \cdot y + x \cdot \bar{y}$

## XI. COMPLEMENTE INFORMATICE



Tabelul 22.7 (continuare)

$f_n$ $x_y$	00	01	10	11	Denumirea funcției	Forma canonică standardizată a funcției
$f_7$	0	1	1	1	Funcția sumă logică „SAU” („sau inclusiv” disjuncția, reuniunea)	$f_7 = x + y$ (sau „ $x \vee y$ ”); $f_7 = \bar{x} \cdot y + x \cdot y + x \cdot \bar{y}$
$f_8$	1	0	0	0	Funcția NICI, funcția „SAU NU” (antidisjuncția, SAU NEGAT, f. lui Peirce)	$f_8 = x \downarrow y$ ; $f_8 = \overline{x + y} = \bar{x} \cdot \bar{y}$
$f_9$	1	0	0	1	Funcția echivalență (identitatea logică, coincidența)	$f_9 = \bar{x} \cdot \bar{y} + x \cdot y$ (sau $x \equiv y$ )
$f_{10}$	1	0	1	0	Funcția complement $y$ (negația lui $y$ )	$f_{10} = \bar{y}$
$f_{11}$	1	0	1	1	Funcție implicație	$f_{11} = \bar{x} \cdot \bar{y} + x \cdot \bar{y} + x \cdot y$ (sau $f_{11} = x + \bar{y}$ , sau $y \rightarrow x$ )
$f_{12}$	1	1	0	0	Funcția complement $x$ (negația lui $x$ )	$f_{12} = \bar{x}$
$f_{13}$	1	1	0	1	Funcția implicație (implicația)	$f_{13} = \bar{x} \cdot \bar{y} + \bar{x} \cdot y + x \cdot y$ (sau $f_{13} = \bar{x} + y$ , sau $x \rightarrow y$ )
$f_{14}$	1	1	1	0	Funcția „ȘI NU”. Anticonjuncția, excluziunea, intersecția inversă, funcția lui SHEFFER	$f_{14} = x \mid y$ ; $f_{14} = \bar{x} \cdot \bar{y} + \bar{x} \cdot y + x \cdot \bar{y}$ $f_{14} = \bar{x} \cdot \bar{y} = \overline{x + y}$
$f_{15}$	1	1	1	1	Funcția constantă „1” (constanta „1”)	$f_{15} = 1$

4. Orice funcție booleană poate fi generată de una sau mai multe expresii boolene. De exemplu, adunarea booleană poate fi generată și de expresia  $E = \bar{x}_1 \cdot x_2 + x_1 \cdot \bar{x}_1 + x_1 \cdot x_2$ . Într-adevăr, folosind proprietățile boolene, rezultă:

$$\begin{aligned}
 \bar{x}_1 \cdot x_2 + x_1 \cdot \bar{x}_2 + x_1 \cdot x_2 &= \bar{x}_1 \cdot x_2 + x_1 \cdot \bar{x}_2 + x_1 \cdot x_2 + x_1 \cdot x_2 && \text{— idempotența} \\
 &= (\bar{x}_1 \cdot x_2 + x_1 \cdot x_2) + (x_1 \cdot \bar{x}_2 + x_1 \cdot x_2) && \text{— comutativitatea și asociativitatea} \\
 &= (\bar{x}_1 + x_1) \cdot x_2 + x_1 \cdot (\bar{x}_2 + x_2) && \text{— distributivitatea} \\
 &= 1 \cdot x_2 + x_1 \cdot 1 && \text{— principiul terțiului exclus} \\
 &= x_1 + x_2.
 \end{aligned}$$

Funcția  $f_9$  este complementul sumei disjunctive. Într-adevăr:

$$\begin{aligned}
 \overline{x \oplus y} &= \overline{(\bar{x} \cdot y + x \cdot \bar{y})} = (\overline{\bar{x} \cdot y}) \cdot (\overline{x \cdot \bar{y}}) = (x + \bar{y}) \cdot (\bar{x} + y) = x \cdot \bar{x} + x \cdot y + \bar{y} \cdot \bar{x} + \bar{y} \cdot y = 0 + x \cdot y + \bar{y} \cdot \bar{x} + 0 = \bar{x} \cdot \bar{y} + x \cdot y
 \end{aligned}$$



Expresia funcției implicație,  $f_{13} = \bar{x} \cdot \bar{y} + x \cdot y + x \cdot y$ , se simplifică și rezultă:

$$\bar{x} \cdot \bar{y} + x \cdot y + x \cdot y = \bar{x} \cdot (\bar{y} + y) + x \cdot y = \bar{x} + x \cdot y = (\bar{x} + x)(\bar{x} + y) = \bar{x} + y$$

Se observă că din cele 16 funcții boolene de două variabile, 8 sînt simetrice:  $f_0, f_1, f_6, f_7, f_8, f_9, f_{14}$ , și  $f_{15}$  (ele se conservă dacă  $x$  și  $y$  permută între ele). De asemenea, fiecare funcție este complementul alteia, respectîndu-se regula:  $f_i = \bar{f}_{15-i}$  ( $i \in \{0, 1, \dots, 15\}$ ).

5. La o analiză mai amănunțită se poate arăta că orice funcție booleană poate fi reprezentată:

- fie numai prin funcțiile de bază (conjuncția, disjuncția și negația);
- fie numai prin  $f_1$  și  $f_{12}$ , sau prin  $f_7$  și  $f_{12}$ ;
- fie numai prin funcția antidisjuncție „ $f_8$ ” a lui Peirce (simbolizată „ $\downarrow$ ”);
- fie numai prin funcția anticonjuncție „ $f_{14}$ ” a lui Sheffer (simbolizată „ $|$ ”).

## 22.4. Forme canonice ale funcțiilor boolene

Așa cum se știe, există mai multe expresii boolene care reprezintă aceeași funcție. Se pune deci problema găsirii unor forme standard de reprezentare prin expresii a unei funcții boolene. În acest mod funcțiile boolene se pot compara între ele comparînd expresiile standard asociate lor. Formele standard de reprezentare a funcțiilor boolene sînt formele normale și formele canonice.

### 4.1. Forme normale (definiții și exemple)

#### 1. Produs elementar

Se numește produs elementar un produs de variabile boolene sau complemente ale acestora, fără ca aceeași variabilă să apară de mai multe ori sau împreună cu complementul său.

De exemplu, „ $\bar{x}yz$ ” este un produs elementar.

Observație. Produsul elementar este zero dacă conține simultan variabile și negația sa ( $a \cdot \bar{a} = 0$ ).

#### 2. Sumă elementară

Se numește sumă elementară o sumă de variabile boolene sau de complemente ale acestora, fără ca aceeași variabilă să apară de mai multe ori sau împreună cu complementul său.

De exemplu, „ $x + \bar{y} + z$ ” este o sumă elementară.

Observație. Suma elementară care conține simultan variabila și complementara sa este egală cu 1 (adică  $a + \bar{a} = 1$ ).

#### 3. Formă normală disjunctivă

Se numește formă normală disjunctivă a unei expresii boolene o sumă de produse elementare egală cu expresia dată.



De exemplu, expresia  $\varepsilon = \bar{a} \cdot \bar{b} + \bar{c} \cdot \bar{d}$  este o formă normală disjunctivă a expresiei  $(a+b) \cdot (c+d)$ , deoarece:

$$(a+b) \cdot (c+d) = (\overline{a+b}) + (\overline{c+d}) = \bar{a} \cdot \bar{b} + \bar{c} \cdot \bar{d}$$

Spunem că o funcție booleană este dată sub forma normală disjunctivă, dacă expresia sa este sub forma normală disjunctivă. Forma normală disjunctivă a unei funcții nu este unică. De exemplu, funcția  $f(a,b,c) = (a+b)(b+c)(c+a)$  este reprezentată de expresiile:

$$\alpha = abc + ac + bc + \bar{a}\bar{b} + \bar{a}\bar{b}c, \quad \beta = ac + bc + \bar{a}\bar{b}.$$

Într-adevăr, avem:

$$\begin{aligned} f(a,b,c) &= (a+b)(b+c)(c+a) = abc + a\bar{a}b + ac + a\bar{a}c + bc + \bar{a}b + bc + \bar{a}bc = \\ &= abc + ac + bc + \bar{a}b + \bar{a}bc = (ac + abc) + bc + (\bar{a}b + \bar{a}bc) = ac + bc + \bar{a}b \end{aligned}$$

Pentru aducerea unei funcții boolene la forma normală disjunctivă se procedează astfel:

- i) se trece complementarea de la expresii la variabilele componente, conform formulelor lui De Morgan (ori de câte ori există expresii complementate);
- ii) se aplică proprietatea de distributivitate a operației „ $\cdot$ ” față de operația „ $+$ ” și proprietatea de asociativitate (ori de câte ori este cazul);
- iii) variabilele, sau produsele, care se repetă se reduc la o singură variabilă, sau la un singur produs (conform idempotenței);
- iv) se elimină din sumă produsul în care aceeași variabilă apare împreună cu complementul său;
- v) după caz, se aplică proprietatea absorbției etc.

#### 4. Forma normală conjunctivă

Se numește formă normală conjunctivă a unei expresii boolene un produs de sume elementare egal cu expresia dată.

De exemplu, expresia  $(a+b)(a+c)(c+\bar{a})$  este o formă normală conjunctivă a expresiei  $\alpha = ac + bc + \bar{a}b$ .

Pentru a arăta că două expresii  $\alpha = a(\bar{a}+b) + \bar{a}(a+b)$  și  $\beta = \bar{a}(b+a) + b(a+\bar{b})$  au o aceeași formă normală conjunctivă se procedează astfel:

- i) se aplică formulele lui De Morgan (dacă este cazul);
- ii) se aplică distributivitatea operației „ $+$ ” față de operația „ $\cdot$ ”;
- iii) se aplică idempotența, se elimină sumele care au valoarea constantă 1 etc.

Astfel, avem:

$$\begin{aligned} \alpha &= a \cdot (\bar{a}+b) + \bar{a} \cdot (a+b) - \text{conform distributivității „}\varepsilon + \mu \nu = (\varepsilon + \mu) \cdot (\varepsilon + \nu)\text{”, unde } \varepsilon = a \cdot (\bar{a}+b), \mu = \bar{a} \text{ și } \nu = a+b, \text{ rezultă} \\ &= (a \cdot (\bar{a}+b) + \bar{a}) \cdot (a \cdot (\bar{a}+b) + a+b) - \text{se ordonează, se aplică de două ori distributivitatea și rezultă} \\ &= (\bar{a} + a \cdot (\bar{a}+b)) \cdot ((a+b) + a \cdot (\bar{a}+b)) = (\bar{a} + a)(\bar{a} + \bar{a} + b)(a+b+a)(a+b+\bar{a}+b) = (a+\bar{a})(\bar{a}+b)(a+b)(a+\bar{a}+b) = 1 \cdot (a+b) \cdot (\bar{a}+b) \cdot (1+b) = \\ &= (a+b) \cdot (\bar{a}+b), \text{ unde } b+1=1. \end{aligned}$$



La fel,  $\beta = (a+b) \cdot (\bar{a}+\bar{b})$ .

## 5. Mintermen

Se numește mintermen (sau minterm) de  $n$  variabile  $x_1, x_2, \dots, x_n$  un produs elementar în care apar toate variabilele simple sau complementate.

Exemple:  $a \cdot b \cdot c, \bar{a} \cdot b \cdot \bar{c}, x_1 \cdot \bar{x}_2 \cdot x_3, x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \cdot \bar{x}_5 \cdot x_6$ .

Oricărui mintermen de variabile  $x_1, x_2, \dots, x_n$  i se poate asocia un număr în baza 2 — obținut prin înlocuirea variabilelor complementate cu cifra „0” și a celor necomplementate cu cifra „1”. Astfel, mintermenilor dați mai sus li se asociază, respectiv, numerele:  $(111)_2, (010)_2, (101)_2, (100101)_2$ .

Correspondentele în baza 10 ale acestor numere sînt, respectiv, numerele: 7, 2, 5, 37.

Cu  $n$  variabile boolene se pot construi  $2^n$  mintermeni diferiți (acesta fiind numărul de numere binare de cîte  $n$  cifre). Dintre toți acești mintermeni diferiți, numai unul are valoarea 1 (atunci cînd fiecare variabilă are valoarea 1).

## 6. Maxtermen

Se numește maxtermen (sau maxterm) de variabilele  $x_1, x_2, \dots, x_n$  o sumă elementară în care apar toate variabilele, simple sau complementate.

Exemple:  $a+\bar{b}+c, \bar{a}+b+c, x+y, x_1+\bar{x}_2+x_3+\bar{x}_4+x_5$  etc.

Oricărui maxtermen de  $n$  variabile  $x_1, x_2, \dots, x_n$  i se poate asocia un număr în baza 2 — obținut prin înlocuirea variabilelor complementate cu „1” și a celor necomplementate cu „0”. Numărul maxtermenilor construiți cu  $n$  variabile este egal cu  $2^n$ . Dintre toți acești maxtermeni diferiți, numai unul are valoarea 0. Pentru stabilirea numărului de mintermeni (respectiv de maxtermeni) ai unei funcții boolene se va analiza funcția definită prin tabela 22.8.

Tabela 22.8

	a	b	c	f(a, b, c)
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

Liniiile acestei tabele sînt numerotate de la 0 la 7 (de la 0 la  $2^n-1$ ,  $n$  fiind numărul variabilelor boolene ale funcției). Valorile variabilelor  $a, b, c$  reprezintă cifre binare. Se observă că fiecărui număr de  $n$  cifre (în cazul de față  $n=3$ ) îi corespunde pe fiecare din cele opt linii cîte un mintermen (respectiv maxtermen) format astfel încît să conțină variabilele din dreptul cifrelor „1” și complementele variabilelor din dreptul cifrelor „0”. De exemplu, pe linia cu numărul 5 există numărul  $(101)_2$ , căruia îi corespunde mintermenul  $m_5 = a \cdot \bar{b} \cdot c$  și maxtermenul  $M_5 = a + \bar{b} + c$ ; pe linia cu numărul 1 există numărul  $(001)_2$ , căruia îi corespunde mintermenul  $m_1 = \bar{a}\bar{b}c$  și maxtermenul  $M_1 = \bar{a} + \bar{b} + c$  etc.



Noțiunea de mintermen este folosită la definirea formei canonice disjunctive a unei expresii boolene, iar noțiunea de maxtermen la definirea formei canonice conjunctive.

## 7. Forma canonică disjunctivă

Se numește formă canonică disjunctivă (sau formă normală disjunctivă perfectă) a unei expresii boolene cu  $n$  variabile o formă normală disjunctivă echivalentă cu expresia, alcătuită numai din mintermeni cu  $n$  variabile.

De exemplu, funcția  $f(a, b, c) = \bar{a} \cdot b \cdot \bar{c} + \bar{a} \cdot b \cdot c + a \cdot \bar{b} \cdot c + a \cdot b \cdot c$  este scrisă sub forma normală disjunctivă perfectă.

Forma normală disjunctivă perfectă a unei funcții boolene de  $n$  variabile este generată de expresia:

$$f_0 \cdot m_0 + f_1 \cdot m_1 + f_2 \cdot m_2 + \dots + f_i \cdot m_i + \dots + f_{2^n-1} \cdot m_{2^n-1}$$

Dacă în exemplul din tabelul 22.8 notăm cu  $m_i$ ,  $i=0,7$ , mintermenii respectivi și cu  $f_i$  valorile funcției  $f$  pentru diversele combinații de valori ale variabilelor  $a, b, c$ :

$$f_0 = f(0,0,0), f_1 = f(0,0,1), f_2 = f(0,1,0), \dots, f_7 = f(1,1,1),$$

atunci forma canonică disjunctivă a funcției  $f(a, b, c)$  este generată de expresia

$$f_0 \cdot m_0 + f_1 \cdot m_1 + f_2 \cdot m_2 + f_3 \cdot m_3 + f_4 \cdot m_4 + f_5 \cdot m_5 + f_6 \cdot m_6 + f_7 \cdot m_7$$

Înlocuind, rezultă:

$$\begin{aligned} & f(0,0,0) \cdot \bar{a} \cdot \bar{b} \cdot \bar{c} + f(0,0,1) \cdot \bar{a} \cdot \bar{b} \cdot c + f(0,1,0) \cdot \bar{a} \cdot b \cdot \bar{c} + f(0,1,1) \cdot \bar{a} \cdot b \cdot c + \\ & + f(1,0,0) \cdot a \cdot \bar{b} \cdot \bar{c} + f(1,0,1) \cdot a \cdot \bar{b} \cdot c + f(1,1,0) \cdot a \cdot b \cdot \bar{c} + f(1,1,1) \cdot a \cdot b \cdot c = \\ & = 0 \cdot \bar{a} \cdot \bar{b} \cdot \bar{c} + 0 \cdot \bar{a} \cdot \bar{b} \cdot c + 1 \cdot \bar{a} \cdot b \cdot \bar{c} + 1 \cdot \bar{a} \cdot b \cdot c + 0 \cdot a \cdot \bar{b} \cdot \bar{c} + 1 \cdot a \cdot \bar{b} \cdot c + \\ & + 0 \cdot a \cdot b \cdot \bar{c} + 1 \cdot a \cdot b \cdot c = \bar{a} \cdot b \cdot \bar{c} + \bar{a} \cdot b \cdot c + a \cdot \bar{b} \cdot c + a \cdot b \cdot c \end{aligned}$$

Observații. 1. Forma canonică disjunctivă a unei funcții boolene se poate obține și direct, pe baza tabelii de valori a funcției, astfel:

i) se aleg din tabelă numai acele linii pentru care funcția  $f$  ia valoarea 1 și se consideră mintermenul corespunzător fiecărei linii alese. (În exemplul dat se iau mintermenii liniilor 2, 3, 5, 7:  $\bar{a} \cdot b \cdot \bar{c}$ ,  $\bar{a} \cdot b \cdot c$ ,  $a \cdot \bar{b} \cdot c$ ,  $a \cdot b \cdot c$ ).

ii) se formează disjuncția mintermenilor obținuți

$$\bar{a} \cdot b \cdot \bar{c} + \bar{a} \cdot b \cdot c + a \cdot \bar{b} \cdot c + a \cdot b \cdot c$$

obținându-se forma normală disjunctivă perfectă a funcției  $f$ .

2. Forma normală disjunctivă a unei funcții poate fi transformată într-o formă normală disjunctivă perfectă înmulțind fiecare termen al formei normale disjunctive, care nu este mintermen, cu expresii de forma „ $x + \bar{x}$ ”, pentru fiecare variabilă  $x$  absentă din acel termen.

Spre exemplu, procedeul de transformare a funcției  $f(a, b, c) = \bar{a} \cdot b + a \cdot c$  este următorul:

$$\begin{aligned} f(a, b, c) &= \bar{a} \cdot b + a \cdot c = \bar{a} \cdot b(c + \bar{c}) + a \cdot c(b + \bar{b}) = \bar{a} \cdot b \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot c + \\ &+ a \cdot b \cdot c \end{aligned}$$



## 8. Forma canonică conjunctivă

Se numește formă canonică conjunctivă, sau formă normală conjunctivă perfectă, o funcție booleană generată de expresia:

$$(f_0 + M_{2^n-1}) \cdot (f_1 + M_{2^n-2}) \cdot (f_2 + M_{2^n-3}) \cdot \dots (f_k + M_{2^n-k-1}) \dots (f_{2^n-1} + M_0)$$

unde  $M_i$  sînt maxtermeni,  $0 \leq k \leq 2^n - 1$ .

Pentru exemplul descris în figura 22.8, forma canonică conjunctivă a funcției  $f(a, b, c)$  este generată de expresia:

$$\begin{aligned} & (f(0,0,0) + M_7) \cdot (f(0,0,1) + M_6) \cdot (f(0,1,0) + M_5) \cdot (f(0,1,1) + M_4) \cdot (f(1,0,0) + \\ & + M_3) \cdot (f(1,0,1) + M_2) \cdot (f(1,1,0) + M_1) \cdot (f(1,1,1) + M_0) = (0 + M_7) \cdot (0 + M_6) \cdot \\ & \cdot (1 + M_5) \cdot (1 + M_4) \cdot (0 + M_3) \cdot (1 + M_2) \cdot (0 + M_1) \cdot (1 + M_0) = (a + b + c) \cdot \\ & \cdot (a + b + \bar{c}) \cdot 1 \cdot 1 \cdot (\bar{a} + b + c) \cdot 1 \cdot (\bar{a} + \bar{b} + c) \cdot 1 = (a + b + c) \cdot (a + b + \bar{c}) \cdot \\ & \cdot (\bar{a} + b + c) \cdot (\bar{a} + \bar{b} + c). \end{aligned}$$

Observații. 1. Forma canonică disjunctivă și cea conjunctivă constituie reprezentări standard ale funcțiilor boolene — aceste forme avînd proprietatea de unicitate.

2. Pentru aflarea formei canonice conjunctive a funcției  $f(a,b,c)$ , cu ajutorul tabelui, se iau în considerație toate liniile pentru care  $f(a,b,c) = 0$ .

În fiecare triplet al acestor linii se asociază lui „0” litera corespunzătoare din capul coloanei, iar lui „1” negația literei din capul coloanei, făcîndu-se apoi suma elementelor asociate.

## 22.5. Simplificarea funcțiilor boolene

Există mai multe căi de simplificare a funcțiilor boolene:

- simplificarea prin calcul boolean (metodă algebrică);
- simplificarea pe baza reprezentării sub formă grafică a relațiilor, folosind diagramele Euler-Venn, sau diagramele Veitch-Karnaugh (metodă tabulară).

22.5.1. Simplificarea prin calcul boolean constă în aducerea unei expresii la cea mai simplă formă normală disjunctivă echivalentă cu ea. Pentru aceasta se folosesc proprietățile algebrei Boole: idempotența ( $a + \bar{a} = a$ ,  $a \cdot a = a$ ); absorbția ( $a + a \cdot b = a$ ,  $a \cdot (a + b) = a$ ); terțiul exclus ( $a + \bar{a} = 1$ ), necontradicția ( $a \cdot \bar{a} = 0$ ) și alte reguli de combinare ( $a \cdot b + a \cdot \bar{b} = a$ ,  $(a + b) \cdot (a + \bar{b}) = a$ ) etc.

În procesul simplificării sînt uneori necesare artificii de calcul, de exemplu — introducerea unor termeni sau factori noi, pentru a se facilita unele simplificări ulterioare (mai ales cînd se aplică absorbția sau idempotența).

Aplicație. În simplificarea expresiei  $\alpha = a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c + a \cdot b$  avem etapele:

$$\begin{aligned} \alpha &= a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c + a \cdot b = a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c + a \cdot b + a \cdot \bar{b} \cdot c \text{ (absorbția} \\ & \text{„}x + xy = x\text{”)} \end{aligned}$$



$$\begin{aligned}
&= a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c + a \cdot b + a \cdot b \cdot c + a \cdot b \cdot c \text{ (idempotența „} x+x=x \text{)"} \\
&= a \cdot b \cdot c + a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c + a \cdot b \cdot c + a \cdot b = a \cdot c(b + \bar{b}) + b \cdot c(\bar{a} + a) + \\
&\quad + a \cdot b = \\
&= a \cdot b + a \cdot c + b \cdot c.
\end{aligned}$$

Observație. Pentru simplificarea funcțiilor boolene este utilă și noțiunea de termen canonic P (termen de produse, sau termen P). Acest termen este format din produsul logic al celor  $n$  variabile (simple sau complementate) ale unei funcții boolene, cu condiția ca produsul să aibă valoarea „1”. De asemenea, este utilă și noțiunea de termen canonic S (sau termen sume), care este format din suma logică a celor  $n$  variabile (simple sau complementate) ale unei funcții boolene, cu condiția ca această sumă să aibă valoarea logică „0”.

Termenii P și S ai funcțiilor boolene de două și de trei variabile sînt dați în tabelele 22.9-a și 22.9-b.

Tabelul 22.9-a

a	b	$\bar{a}$	$\bar{b}$	Termeni P	Termeni S
0	0	1	1	$\bar{a} \cdot \bar{b} = 1$	$a + b = 0$
0	1	1	0	$\bar{a} \cdot b = 1$	$a + \bar{b} = 0$
1	0	0	1	$a \cdot \bar{b} = 1$	$\bar{a} + b = 0$
1	1	0	0	$a \cdot b = 1$	$\bar{a} + \bar{b} = 0$

Tabelul 22.9-b

a	b	c	$\bar{a}$	$\bar{b}$	$\bar{c}$	Termeni P	Termeni S
0	0	0	1	1	1	$\bar{a} \cdot \bar{b} \cdot \bar{c} = 1$	$a + b + c = 0$
0	0	1	1	1	0	$\bar{a} \cdot \bar{b} \cdot c = 1$	$a + b + \bar{c} = 0$
0	1	0	1	0	1	$\bar{a} \cdot b \cdot \bar{c} = 1$	$a + \bar{b} + c = 0$
0	1	1	1	0	0	$\bar{a} \cdot b \cdot c = 1$	$a + \bar{b} + \bar{c} = 0$
1	0	0	0	1	1	$a \cdot \bar{b} \cdot \bar{c} = 1$	$\bar{a} + b + c = 0$
1	0	1	0	1	0	$a \cdot \bar{b} \cdot c = 1$	$\bar{a} + b + \bar{c} = 0$
1	1	0	0	0	1	$a \cdot b \cdot \bar{c} = 1$	$\bar{a} + \bar{b} + c = 0$
1	1	1	0	0	0	$a \cdot b \cdot c = 1$	$\bar{a} + \bar{b} + \bar{c} = 0$

### 22.5.2. Simplificarea pe baza diagramelor Euler-Venn

Într-o diagramă Euler-Venn, fiecare variabilă a unei expresii boolene este reprezentată printr-o mulțime (sub forma unui cerc — al cărui interior corespunde variabilei, iar exteriorul corespunde complementului variabilei). În acest mod, un termen al expresiei boolene corespunde intersecțiilor mulțimilor corespunzătoare variabilelor ce-l compun, iar expresia corespunde reuniunii mulțimilor corespunzătoare termenilor componenți. Vom nota mulțimea corespunzătoare unei variabile prin majusculă (de exemplu, variabilei  $a$  — mulțimea  $A$ , variabilei  $b$  — mulțimea  $B$  etc.).



Astfel, diagramele din figurile 22.6 (a,b,c) reprezintă, respectiv, una, două sau trei variabile și mulțimile corespunzătoare zonelor acestor variabile.

Analizînd diagrama 22.6-c, se observă că mintermenii:

$m_7=abc$ ,  $m_5=\bar{a}bc$ ,  $m_4=a\bar{b}c$  și  $m_6=ab\bar{c}$  corespund zonelor a căror reuniune formează mulțimea A. Deci, avem relația:

$$a \cdot b \cdot c + a \cdot \bar{b} \cdot c + a \cdot \bar{b} \cdot \bar{c} + a \cdot b \cdot \bar{c} = a$$

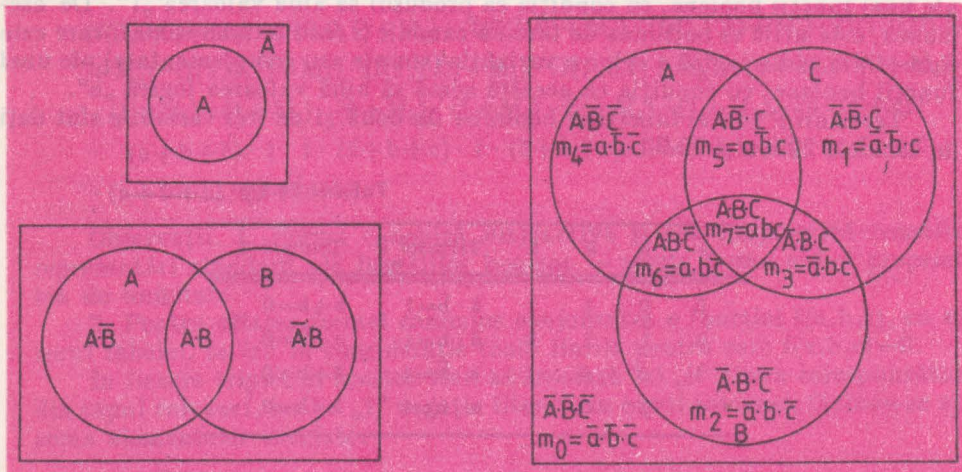


Fig. 22.6. a) — Diagramă cu o variabilă; b) — Diagramă cu 2 variabile; c) — Diagramă cu trei variabile

În acest mod, conform figurilor 22.6 (b, c), rezultă următoarele simplificări:

$$\begin{aligned} a \cdot \bar{b} + a \cdot b &= a; & a \cdot b + \bar{a} \cdot b &= b; & a \cdot \bar{b} + a \cdot b + \bar{a} \cdot b &= a + b; & a \cdot b \cdot c + a \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot c + \\ &+ a \cdot \bar{b} \cdot \bar{c} &= a; & a \cdot b \cdot c + a \cdot b \cdot \bar{c} + \bar{a} \cdot b \cdot c + \bar{a} \cdot b \cdot \bar{c} &= b; & a \cdot b \cdot c + a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c + \\ &+ \bar{a} \cdot \bar{b} \cdot c &= c \text{ etc.} \end{aligned}$$

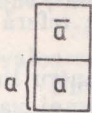
Evident, aceste simplificări se pot realiza și prin metoda algebrică, pe baza calculului boolean. Desigur, reprezentarea geometrică de mai sus devine greoaie dacă expresiile conțin mai mult de trei variabile.

### 22.5.3. Simplificarea pe baza diagramelor Veitch-Karnaugh („V.—K“)

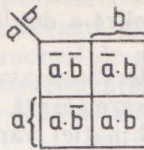
Diagramele Veitch-Karnaugh se prezintă sub forma unui tablou pătratic sau dreptunghiular. Dacă transformăm diagramele Euler-Venn din figurile 22.6 (a, b, c) în tablouri pătratice sau dreptunghiulare, în care mulțimile care reprezintă variabilele boolene sînt fășii dreptunghiulare nedisjuncte, obținem diagrame de tip Veitch-Karnaugh pentru una, două sau trei variabile, ca în tabelele 22.10 (a, b, c, d). Prin intermediul acestor diagrame se pot pune mai ușor în evidență termenii de forma  $Xa + X\bar{a}$ , care se simplifică obținînd  $X$ .



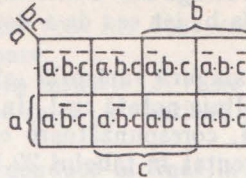
Tabelul 22.10-a



Tabelul 22.10-b



Tabelul 22.10-c



Tabelul 22.10-d

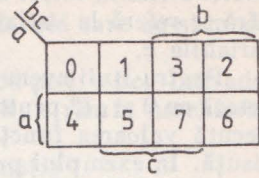


Fig. 22.7 a) — Diagrama V—K pentru o variabilă; b) — Diagrama V—K pentru două variabile; c) — Diagrama V—K pentru o funcție de trei variabile; d) — Diagrama d

Fie, pentru reprezentare, diagrama unei funcții de trei variabile — un tablou format din două linii și patru coloane căruia i s-au numerotat căsuțele ca în tabelul 22.10-d. Pentru o funcție logică de trei variabile există  $2^3=8$  variante de combinare a valorilor variabilelor, ea conținând deci cel mult opt termeni. În fiecare căsuță se scrie valoarea pe care o ia funcția pentru ansamblul de valori ale variabilelor. Acest ansamblu de valori se găsește urmărind coordonatele căsuței (pătrățului).

Spre exemplificare, fie funcția  $f(a, b, c)$ , prezentată tabelar în 22.11-a.

Tabela 22.11-a

a	b	c	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Tabela 22.11-b

a/b	00	01	11	10	
0					linia „0”
1					linia „1”
	coloana „0”	coloana „1”	coloana „3”	coloana „2”	

Tabela 22.11-d

a/b	0	1	3	2
0				
1				

Tabela 22.11-c

În tabela 22.11-b se pot evidenția coordonatele căsuțelor. Conform tablei 22.11-c, se consideră că a, b, c au valoarea 1 logic în zonele indicate de acolade. Astfel, luând coordonatele căsuțelor, se constată că pentru căsuța numerotată cu 3 avem mintermenul  $\bar{a}bc$  — deci tripletul ( $a=0, b=1, c=1$ ); pentru căsuța numerotată cu 7, avem mintermenul  $abc$  — deci tripletul ( $a=1, b=1, c=1$ ) etc. În căsuța numerotată cu 3 va fi trecut valoarea funcției — corespunzător combinației  $a=0, b=1, c=1$ , deci  $f=1$  (conform tablei de adevăr 22.11-a); în căsuța 7 va fi trecut valoarea funcției pentru  $a=1, b=1, c=1$ , deci  $f=1$ .

Dacă se renunță la reprezentarea variabilelor cu acolade și se scrie numai valoarea variabilei corespunzătoare fiecărei linii sau fiecărei coloane (tabelul 22.11-d), se observă că:  $b=0$  pentru primele două coloane (coloana „0” și coloana „1”) și  $b=1$  pentru celelalte două coloane;  $c=0$  pentru prima și ultima coloană („0” și „2”) și  $c=1$  pentru coloanele din mijloc („1” și „3”);  $a=0$  în linia notată cu „0” și  $a=1$  în linia notată cu „1”. Scriem aceste valori ale variabi-



ei în dreptul liniilor (la stînga) și deasupra coloanelor, respectînd ordinea variaa-  
bilelor scrise în colțul din stînga-sus. Conform tabelului 22.11-d, pe coloane, prima  
cifră se referă la variabila  $b$ , iar cea de-a doua cifră a dubletului se referă la  
variabila  $c$ .

Pentru linia avem o singură variabilă „ $a$ ”, deci vom avea „0” pentru linia  
notată cu 0 și „1” pentru linia notată cu 1. În fiecare căsuță a diagramei va fi  
trecută valoarea funcției, corespunzătoare combinației variabilelor din acea  
căsuță. În exemplul prezentat în tabelul 22.11-a se observă că  $f$  ia valoarea 1  
pentru combinațiile notate cu „\*” în tabelă. Aceste valori sînt trecute în tabe-  
lă 22.11-e, astfel:

Tabelul 22.11-e

$\begin{smallmatrix} a \\ b/c \end{smallmatrix}$	00	01	11	10
0	0	0	1	0
1	1	1	1	0

Tabelul 22.11-f

$\begin{smallmatrix} a \\ b/c \end{smallmatrix}$	0	0	1	0
0	0	0	1	0
1	1	1	1	0

Fig. 22.8. a) — Reprezentare tabelară; b) — Diagrama b; c) — Diagrama c; d) — Diagrama  
d; e) — Grupări de căsuțe; f) — Simplificare

Prima valoare „1” corespunde, pe linia I, pentru tripletul „ $\bar{a}bc$ ” dat de  
„011” în tabela 22.11-a și se găsește urmărind coloana în care lui  $bc$  îi cores-  
punde „11” și linia în care lui  $a$  îi corespunde „0”. La intersecția acestora (că-  
suța „3”) vom scrie valoarea „1”. La fel, pentru tripletul (100), din tabela 22.11-  
a, avem  $f=1$  și vom înscrie această valoare la intersecția coloanei corespunză-  
toare dubletului (00) cu linia corespunzătoare lui 1, deci în căsuța 4. În conti-  
nuare se obțin căsuțele 5 și 7 pentru care  $f=1$ . Pentru celelalte căsuțe rămase  
 $f$  ia valoarea „0” pe care — eventual — nu o mai trecem în diagramă. Putem  
scrie algebraic funcția  $f$  ca fiind suma tuturor termenilor pentru care aceasta ia  
valoarea „1”, deci:

$$f = \bar{a}bc + a\bar{b}c + ab\bar{c} + abc$$

Simplificarea algebrică (booleană) a funcției  $f$  constă în gruparea convena-  
bilă a termenilor astfel încît să se poată evidenția termeni de forma „ $X.a + X.\bar{a}$ ”,  
respectiv factori de forma „ $(x + \bar{x})$ ”, a a căror valoare este  $X$ , respectiv 1. Anali-  
zînd diagrama din tabelul 22.10-c, se constată că fiecare pătrat reprezintă o  
intersecție a tuturor mulțimilor corespunzătoare variabilelor boolene sau com-  
plementelor acestora — deci, fiecare pătrat reprezintă un mintermen. De ase-  
menea, mintermenii care diferă între ei printr-un singur factor sînt reprezentați  
prin pătrate alăturate. Cu alte cuvinte, la două căsuțe alăturate una din varia-  
bile apare negată într-o căsuță și nenegată în cealaltă, restul variabilelor rămî-  
nînd neschimbate. Se observă, de aci, că prin reuniunea a două pătrate vecine  
apar evidente simplificări de tipul  $X.a + X.\bar{a} = X(a + \bar{a}) = X$ .

În consecință, se urmărește formarea unor grupe de câte 2, 4, 8, 16 etc.)  
căsuțe alăturate care-l conțin pe „1”. În diagrama 22.11-e se grupează căsuțele  
(4, 5) și (3, 7), ca în tabelul 22.11-f. Deci,  $f$  va avea forma simplificată  
 $f = t_1 + t_2$ .

În gruparea (4, 5) din diagrama 22.10-c, variabilele  $a$  și  $b$  sînt fixe (nu-și  
modifică valoarea) pe cînd variabila  $c$  își modifică valoarea (adică apare atît



negată cît și nenegată — deci nu mai apare în produs). În consecință, se obține termenul  $t = \bar{a}\bar{b}$ , care rezultă din simplificarea celor doi termeni astfel:  
 $\bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c = \bar{a}\bar{b}(c + \bar{c}) = \bar{a}\bar{b}.1 = \bar{a}\bar{b}$ .

Practic, din această grupare vom lua doar variabilele care nu-și modifică valoarea prin procesul de grupare.

Simplificarea termenilor din căsuțele (3, 7) ni-l dă pe  $t = bc$ . Într-adevăr, avem:  $\bar{a}bc + abc = (\bar{a} + a)bc = bc$  (aici „a” variază și b, c rămîn neschimbați), deci  $t = bc$ .

Simplificarea booleană ne conduce în final la funcția

$$f = bc + \bar{a}\bar{b},$$

care conține tot atîția termeni cîte grupări există. Dacă un termen „1” nu intră în nici o grupare, el va rămîne în forma finală a funcției simplificate (el nu se simplifică).

Observații: i) Simplificarea prin diagrama Veitch-Karnaugh se poate extinde și asupra funcțiilor de 4, 5, 6 variabile.

ii) În tabelul 22.12-a este prezentat un mod de numerotare a căsuțelor diagramelor V-K pentru funcții de patru variabile, iar în tabelele 22.12 (b, c, d, e, f, g) sînt prezentate diferite modele de simplificare:

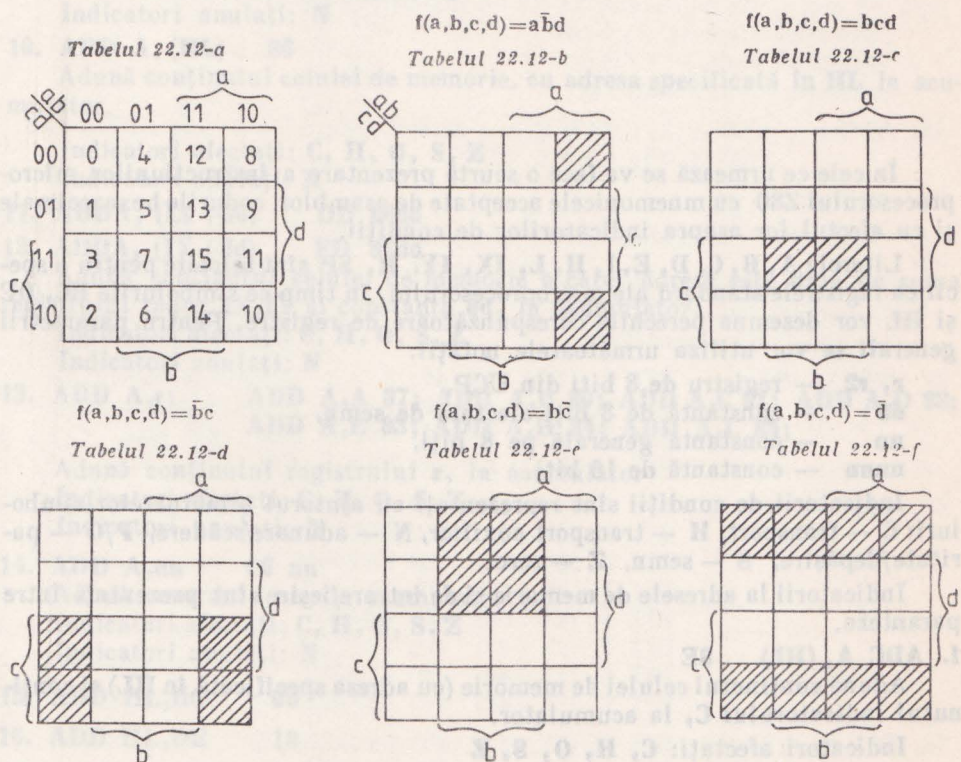
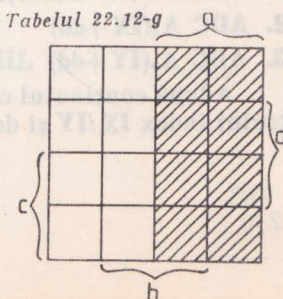


Fig. 22.12. a) — Numerotarea căsuțelor; b) —  $f(a, b, c, d) = \bar{a}\bar{b}d$ ; c) —  $f(a, b, c, d) = bcd$ ; d) —  $f(a, b, c, d) = \bar{b}c$ ; e) —  $f(a, b, c, d) = b\bar{c}$ ; f) —  $f(a, b, c, d) = \bar{d}$ ; g) —  $f(a, b, c, d) = a$

iii) Diagrama V-K poate fi considerată și ca o tabelă de definiție cu două dimensiuni a unei funcții.

iv) Exprimarea unei funcții minime se poate face și prin produse (nu sume), realizîndu-se o sinteză după „zero-uri”, nu după „1” — ca pînă aici.





## COMPLEMENTE INFORMATICE.

### Capitolul 23.

### Instrucțiunile microprocesorului Z 80

În cele ce urmează se va face o scurtă prezentare a instrucțiunilor microprocesorului Z80 cu mnemonicele acceptate de asamblor, codurile hexazecimale și cu efectul lor asupra indicatorilor de condiții.

Literele A, B, C, D, E, I, H, L, IX, IY, R, SP sînt folosite pentru a specifica registrele standard ale microprocesorului, în timp ce simbolurile BC, DE și HL vor desemna perechile corespunzătoare de registre. Pentru parametrii generali se vor utiliza următoarele notații:

- r, r2** — registru de 8 biți din UCP,
- dd** — constantă de 8 biți afectată de semn,
- nn** — constantă generală pe 8 biți,
- nnnn** — constantă de 16 biți.

Indicatorii de condiții sînt reprezentați cu ajutorul următoarelor simboluri: C — transport, H — transport auxiliar, N — adunare/scădere, P/O — paritate/depășire, S — semn, Z — zero.

Indicatorii la adresele de memorie și de intrare/ieșire sînt prezentați între paranteze.

#### 1. ADC A, (HL) 8E

Adună conținutul celulei de memorie (cu adresa specificată în HL) și conținutul indicatorului C, la acumulator.

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N.

#### 2. ADC A, (IX+dd) DD 8Edd

#### 3. ADC A, (IY+dd) FD 8Edd

Adună conținutul celulei de memorie (cu adresa specificată de suma registrului index IX/IY și deplasarea dd) și al indicatorului C, la acumulator.



Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

4. ADC A,r; ADC A,A 8F; ADC A,B 88; ADC A,C 89; ADC A,D 8A;  
ADC A,E 8B; ADC A,H 8C; ADC A,L 8D;

Adună conținutul registrului r și al indicatorului C, la acumulator.

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

5. ADC A,nn CE nn

Adună constanta nn și indicatorul C, la acumulator.

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

6. ADC HL,BC ED 4A

7. ADC HL,DE ED 5A

8. ADC HL,HL ED 6A

9. ADC HL,SP ED 7A

Adună conținutul registrului dublu specificat și conținutul indicatorului C, la registrul dublu HL

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

10. ADD A, (HL) 86

Adună conținutul celulei de memorie, cu adresa specificată în HL la acumulator.

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

11. ADDA, (IX+dd) DD 86dd

12. ADDA, (IY+dd) FD 86dd

Adună conținutul celulei de memorie a cărei adresă este dată de suma registrului index IX/IY și constanta dd, la acumulator.

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

13. ADD A,r; ADD A,A 87; ADD A,B 80; ADD A,C 81; ADD A,D 82;  
ADD A,E 83; ADD A,H 84; ADD A,L 85;

Adună conținutul registrului r, la acumulator

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

14. ADD A,nn C6 nn

Adună constanta nn, la acumulator

Indicatori afectați: C, H, O, S, Z

Indicatori anulați: N

15. ADD HL,BC 09

16. ADD HL,DE 19

17. ADD HL,HL 29

18. ADD HL,SP 39

Adună conținutul registrului dublu specificat, la HL

Indicatori afectați: C, O, S, Z



Indicatori anulați: N

- 19. ADD IX,BC DD 09
- 20. ADD IX,DE DD 19
- 21. ADD IX,IX DD 29
- 22. ADD IX,SP DD 39
- 23. ADD IY,BC FD 09
- 24. ADD IY,DE FD 19
- 25. ADD IY,IY FD 29
- 26. ADD IY,SP FD 39

Adună registrul dublu indicat, la registrul index IX/IY.

Indicatori afectați: C, O, S, Z

Indicatori anulați: N

- 27. AND (HL) A6
- 28. AND (IX+dd) DD A6dd
- 29. AND (IY+dd) FD A6dd

Înmulțește logic acumulatorul cu conținutul celulei de memorie adresată prin HL sau suma registrelor IX/IY cu deplasarea dd.

Indicatori afectați: P, S, Z

Indicatori anulați: C, N

Indicatori poziționați: H

- 30. AND r AND A A7; AND B A0; AND C A1; AND D A2; AND E A3;  
AND H A4; AND L A5;

- 31. AND nn E6 nn

Înmulțește logic acumulatorul cu conținutul registrului r sau cu constanta nn.

Indicatori afectați: P, S, Z

Indicatori anulați: C, N

Indicatori poziționați: H

- 32. BIT b, (HL) BIT 0, (HL) CB 46; BIT 1, (HL) CB 4E;  
BIT 2, (HL) CB 56; BIT 3, (HL) CB 5E;  
BIT 4, (HL) CB 66; BIT 5, (HL) CB 6E;  
BIT 6, (HL) CB 76; BIT 7, (HL) CB 7E;

- 33. BIT b, (IX+dd); BIT 0, (IX+dd) DD CBdd46;  
BIT 1, (IX+dd) DD CBdd4E;  
BIT 2, (IX+dd) DD CBdd56;  
BIT 3, (IX+dd) DD CBdd5E;  
BIT 4, (IX+dd) DD CBdd66;  
BIT 5, (IX+dd) DD CBdd6E;  
BIT 6, (IX+dd) DD CBdd76;  
BIT 7, (IX+dd) DD CBdd7E;

- 34. BIT b, (IY+dd) BIT 0, (IY+dd) FD CBdd46;  
BIT 1, (IY+dd) FD CBdd4E;  
BIT 2, (IY+dd) FD CBdd56;  
BIT 3, (IY+dd) FD CBdd5E;  
BIT 4, (IY+dd) FD CBdd66;  
BIT 5, (IY+dd) FD CBdd6E;



BIT 6, (IY+dd) FD CBdd76;  
 BIT 7, (IY+dd) FD CBdd7E;  
 35. BITb, r; BIT 0,A CB 47; BIT 0,B CB 40; BIT 0,C CB 41;  
 BIT 1,A CB 4F; BIT 1,B CB 48; BIT 1,C CB 49;  
 BIT 2,A CB 57; BIT 2,B CB 50; BIT 2,C CB 51;  
 BIT 3,A CB 5F; BIT 3,B CB 58; BIT 3,C CB 59;  
 BIT 4,A CB 67; BIT 4,B CB 60; BIT 4,C CB 61;  
 BIT 5,A CB 6F; BIT 5,B CB 68; BIT 5,C CB 69;  
 BIT 6,A CB 77; BIT 6,B CB 70; BIT 6,C CB 71;  
 BIT 7,A CB 7F; BIT 7,B CB 78; BIT 7,C CB 79;  
 BIT 0,D CB 42; BIT 0,E CB 43; BIT 0,H CB 44;  
 BIT 1,D CB 4A; BIT 1,E CB 4B; BIT 1,H CB 4C;  
 BIT 2,D CB 52; BIT 2,E CB 53; BIT 2,H CB 54;  
 BIT 3,D CB 5A; BIT 3,E CB 5B; BIT 3,H CB 5C;  
 BIT 4,D CB 62; BIT 4,E CB 63; BIT 4,H CB 64;  
 BIT 5,D CB 6A; BIT 5,E CB 6B; BIT 5,H CB 6C;  
 BIT 6,D CB 72; BIT 6,E CB 73; BIT 6,H CB 74;  
 BIT 7,D CB 7A; BIT 7,E CB 7B; BIT 7,H CB 7C;  
 BIT 0,L CB 45;  
 BIT 1,L CB 4D;  
 BIT 2,L CB 55;  
 BIT 3,L CB 5D;  
 BIT 4,L CB 65;  
 BIT 5,L CB 6D;  
 BIT 6,L CB 75;  
 BIT 7,L CB 7D;

Testează bitul *b* al octetului de memorie, a cărei adresă este specificată în HL sau IX/IY+dd, sau bitul *b* al registrului *r*.

Indicatorul de zero este poziționat în unu, dacă bitul testat este unu, în caz contrar indicatorul este anulat. Z ia valoarea complementară a bitului *b*.

Indicatori afectați: Z

Indicatori anulați: N

Indicatori poziționați în unu: H.

### 36. CALL nnnn CD nnnn

Chemare necondiționată de subrutină la adresa nnnn. Adresa următoarei instrucțiuni este plasată în stivă.

Indicatori afectați: nici unul.

### 37. CALL C, nnnn DC nnnn

### 38. CALL M, nnnn FC nnnn

### 39. CALL NC, nnnn D4 nnnn

### 40. CALL NZ, nnnn C4 nnnn

### 41. CALL P, nnnn F4 nnnn

### 42. CALL PE, nnnn EC nnnn

### 43. CALL PO, nnnn E4 nnnn

### 44. CALL Z, nnnn CC nnnn

Chemări condiționate de subrutină, de la adresa nnnn. Adresa instrucțiunii următoare este plasată în stivă. Condițiile sint următoarele:



- C** — indicator de transport = 1 (transport prezent; **C**=1).  
**M** — indicator de semn = 1 (rezultat negativ; **S**=1).  
**NC** — indicator de transport = 0 (transport absent; **C**=0).  
**NZ** — indicator de zero = 0 (rezultat diferit de zero; **Z**=0).  
**P** — indicator de semn = 0 (rezultat pozitiv; **S**=0).  
**PE** — indicator de paritate = 1 (paritate para; **P**=1).  
**PO** — indicator de paritate = 0 (paritate impara; **P**=0).  
**Z** — indicator de zero = 1 (rezultat egal cu zero).

#### 45. CCF 3F

Complementează indicatorul de transport

Indicatori afectați: **C**

Indicatori anulați: **N**

#### 46. CP (HL) BE

#### 47. CP (IX+dd) DD BEdd

#### 48. CP (IY+dd) FD BEdd

Compară octetul din memorie, a cărui adresă este specificată în **HL** sau de suma **IX/IY+dd**, cu acumulatorul. Indicatorul **Z** este poziționat în unu, în cazul egalității. Indicatorul **C** este poziționat în unu, dacă conținutul acumulatorului este mai mic decât operandul specificat.

Indicatori afectați: **C, H, O, S, Z**

Indicatori poziționați: **N**.

#### 49. CP r; CP A BF; CP B B8; CP C B9; CP D BA; CP E BB; CP H BC; CP L BD;

#### 50. CP nn FE nn

Compară conținutul registrului **r** sau constanta **nn**, cu conținutul acumulatorului. Indicatorul **Z** este poziționat în unu, în cazul egalității. Indicatorul **C** este poziționat în unu, dacă conținutul acumulatorului este mai mic decât operandul.

Indicatori afectați: **C, H, O, S, Z**

Indicatori poziționați: **n**

#### 51. CPD ED A9

#### 52. CPDR ED B9

#### 53. CPI ED A1

#### 54. CPIR ED B1

Compară octetul de memorie, specificat în **HL**, cu acumulatorul. Decrementează **HL**, pentru **CPD**, sau incrementează **HL**, pentru **CPI**. Repetă operația, cu decrementarea lui **BC**, pentru **CPDR** sau **CPIR**, până când apare egalitatea sau până când conținutul perechii de registre **BC** a devenit egal cu zero. Bitul de paritate este poziționat în unu, dacă **BC** a fost decrementat până la zero.

Indicatori afectați: **H, S**

Indicatori poziționați: **N, Z** — dacă **A** este egal cu octetul indicat de (**HL**),  
**P** — dacă conținutul lui **BC** este egal cu zero.

#### 55. CPL 2F

Complementează acumulatorul (complementul față de unu); inversare.

Indicatori afectați: **H, N**

#### 56. DAA 27



Ajustare zecimală a acumulatorului. Instrucțiunea se folosește după adunarea și scăderea numerelor reprezentate în codul binar-zecimal.

Indicatori afectați: C, H, O, S, Z.

57. DEC (HL) 35

58. DEC (IX+dd) DD 35dd

59. DEC (IY+dd) FD 35dd

Decrementează cuvântul din memorie, indicat de HL sau IX/IY+dd

Indicatori afectați: H, O, S, Z

Indicatori poziționați: N

Indicatori neafectați: C

60. DEC r; DEC A 3D; DEC B 05; DEC C 0D; DEC D 15;  
DEC E 1D; DEC H 25; DEC L 2D;

Decrementează registrul r. Registrul nu se va decrementa până la zero, într-o buclă JP NC, nnnn, deoarece indicatorul de transport nu va fi afectat.

Indicatori afectați: H, O, S, Z

Indicatori poziționați: N

Indicatori neafectați: C

61. DEC BC 0B

62. DEC DE 1B

63. DEC HL 2B

64. DEC SP 3B

65. DEC IX DD 2B

66. DEC IY FD 2B

Decrementează registrul dublu sau registrul index specificate. Registrul dublu nu se va decrementa până la zero, într-o buclă JP NC, nnnn, întrucât indicatorii nu vor fi afectați. Se poate deplasa, în acumulator, un octet al registrului dublu, pentru a se efectua operația logică SAU, cu celălalt octet al registrului respectiv. Operația SAU va poziționa indicatorii de condiții.

67. DI F3

Dezactivează sistemul cererilor de întrerupere mascabile.

68. DJNZ dd 10 dd

Decrementează registrul B și transferă comanda cu deplasarea relativă dd, dacă BC este diferit de zero.

Indicatori afectați: nici unul.

69. EI FB

Activează sistemul cererilor de întrerupere mascabile.

70. EX (SP), HL E3

71. EX (SP), IX DD E3

72. EX (SP), IY FD E3

Interschimb între conținuturile primelor două celule din vârful stivei și registrele de 16 biți: HL, IX, IY.

Indicatori afectați: nici unul.

73. EX AF, AF 08

Interschimb între registrele acumulator/indicator, din setul de bază, cu cele din setul auxiliar.

Indicatori afectați: toți.



**74. EX DE, HL EB**

Interschimb între conținuturile registrelor duble **DE** și **HL**.

Indicatori afectați: nici unul.

**75. EXX D9**

Interschimb între registrele duble **BC, DE, HL**, din setul de bază și cel auxiliar.

Indicatori afectați: nici unul.

**76. HALT 76**

Se suspendă operarea UCP, până la recepționarea unui semnal **RESET** sau de întrerupere.

**77. IM 0 ED 46**

**78. IM 1 ED 56**

**79. IM 2 ED 5E**

Stabilește modurile de întrerupere: 0, 1 sau 2. *Modul 0* se stabilește automat, după **RESET**, rezultând o manieră de tratare a interupерilor mascabile, asemănătoare cu cea a microprocesorului 8080. *Modul 1* asigură efectuarea instrucțiunii **RST 3BH**, la apariția unei cereri de întrerupere. *Modul 2* asigură mai multe locații de întrerupere.

**80. IN r, (C); IN A, (C) ED 78; IN B, (C) ED 40;**

**IN C, (C) ED 40; IN D, (C) ED 50;**

**IN E, (C) ED 58; IN H, (C) ED 60;**

**IN L, (C) ED 68;**

Încarcă în registrul **r** conținutul portului de intrare a cărui adresă este în registrul **C**.

Indicatori afectați: **P, S, Z**

Indicatori anulați: **H, N**

**81. IN A, (nn) DB nn**

Încarcă în **A** conținutul portului de intrare cu adresa **nn**.

Indicatori afectați: nici unul.

**82. INC (HL) 34**

**83. INC (IX+dd) DD 34dd**

**84. INC (IY+dd) FD 34dd**

Incrementează octetul de memorie cu adresa specificată de **HL, IX/IY+dd**.

Indicatori afectați: **H, O, S, Z**.

Indicatori poziționați: **N**

Indicatori neafectați: **C**.

**85. INC r; INC A 3C; INC B 04; INC C 0C; INC D 14;**

**INC E 1C; INC H 24; INC L 2C;**

Incrementează registrul **r**. Registrul nu se va incrementa peste zero, în ciclurile **JP NC, nnnn**, deoarece instrucțiunea nu afectează indicatorul **C**.

Indicatori afectați: **H, O, S, Z**

Indicatori poziționați: **N**

Indicatori neafectați: **C**.

**86. INC BC 03**

**87. INC DE 13**

**88. INC HL 23**

**89. INC SP 33**

**90. INC IX DD 23**



**91. INC IY      FD 23**

Incrementează registrele specificate.

Indicatori afectați: nici unul.

**92. IND            ED AA**

**93. INDR          ED BA**

**94. INI            ED A2**

**95. INIR          ED B2**

Încarcă un octet de la portul de intrare, cu adresa specificată în registrul **C**, în celula de memorie cu adresa specificată în **HL**. Se decrementează conținutul registrului **B**. Registrul **HL** este incrementat. Instrucțiunile **INDR** și **INIR** se vor repeta până când registrul, de 8 biți, **B** devine egal cu zero.

Indicatori afectați **Z** (dacă **B=0**).

Indicatori poziționați: **N**

**96. JP (IIL)      E9**

**97. JP (IX)       DD E9**

**98. JP (IY)       FD E9**

Încarcă conținutul lui **HL**, **IX**, **IY** în contorul programului **PC**, apoi citește instrucțiunea cu această adresă.

Indicatori afectați: nici unul.

**99. JP nnnn      C3 nnnn**

Transfer necondiționat la instrucțiunea cu adresa **nnnn**.

Indicatori afectați: nici unul.

**100. JP C, nnnn   DA nnnn**

**101. JPM, nnnn    FA nnnn**

**102. JP NC, nnnn   D2 nnnn**

**103. JP NZ, nnnn   C2 nnnn**

**104. JP P, nnnn    F2 nnnn**

**105. JP PE, nnnn   EA nnnn**

**106. JP PO, nnnn   E2 nnnn**

**107. JP Z, nnnn    CA nnnn**

Transfer condiționat la instrucțiunea cu adresa **nnnn**.

Indicatori afectați: nici unul.

**108. JR dd,            18 dd**

Transfer necondiționat la instrucțiunea a cărei adresă este egală cu adresa din contorul programului, la care se adaugă deplasarea **dd**, pozitivă sau negativă.

Deplasarea este relativă în limitele: **+127/-128** octeți, în raport cu adresa instrucțiunii **JR**.

Indicatori afectați: nici unul.

**109. JR C, dd            38 dd**

**110. JR NC dd           30 dd**

**111. JR NZ, dd           20 dd**

**112. JR Z, dd            28 dd**

Transfer condiționat, relativ, al comenzii.

Indicatori afectați: nici unul.

**113. LD (BC), A        02**

**114. LD (DE), A        12**



Încarcă octetul din acumulator, în celula de memorie cu adresa specificată în BC sau DE.

Indicatori afectați: nici unul.

115. LD (HL), r; LD (HL), A 77; LD (HL), B 70; LD (HL), C 71;  
LD (HL), D 72; LD (HL), E 73; LD (HL), H 74;  
LD (HL), L 75;

116. LD (HL), nn 36 nn

117. LD (IX+dd), r; LD (IX+dd), A DD 77dd; LD (IX+dd), B DD 70dd;  
LD (IX+dd), C DD 71dd; LD (IX+dd), D DD 72dd;  
LD (IX+dd), E DD 73dd; LD (IX+dd), H DD 74dd;  
LD (IX+dd), L DD 75dd;

118. LD (IX+dd), nn DD 36ddnn

119. LD (IY+dd), r; LD (IY+dd), A FD 77dd; LD (IY+dd), B FD 70dd;  
LD (IY+dd), C FD 71dd; LD (IY+dd), D FD 72dd;  
LD (IY+dd), E FD 73dd; LD (IY+dd), H FD 74dd;  
LD (IY+dd), L FD 75dd;

120. LD (IY+dd), nn FD 36ddnn

Încarcă în memorie conținutul registrului r sau constanta nn, în celula a cărei adresă este specificată de HL sau de IX/IY+dd. Ultimele patru instrucțiuni se pot folosi pentru încărcarea codului binar relocabil.

121. LD (nnnn), A 32 nnnn

Încarcă acumulatorul în celula de memorie cu adresa nnnn.

122. LD (nnnn), BC ED 32nnnn

123. LD (nnnn), DE ED 53nnnn

124. LD (nnnn), HL 22 nnnn

125. LD (nnnn), IX DD 22nnnn

126. LD (nnnn), IY FD 22nnnn

127. LD (nnnn), SP ED 73nnnn

Încarcă octetul mai puțin semnificativ din registrele duble, registrele index sau indicatorul de stivă, în memorie, la locația cu adresa nnnn. Încarcă octetul mai puțin semnificativ în locația cu adresa nnnn+1.

128. LD A, (BC) 0A

129. LD A, (DE) 1A

Încarcă acumulatorul cu octetul din memorie, de la adresa specificată în BC sau DE.

130. LD A, I ED 57

Încarcă acumulatorul cu conținutul registrului vectorului de întrerupere. Indicatorul de paritate reflectă starea bistabilului de activare a întreruperilor.

Indicatori afectați: P, S, Z

Indicatori anulați: H, N

131. LD A, R ED 5F

Încarcă acumulatorul cu conținutul registrului de reîmprospătare. Indicatorul de paritate reflectă starea bistabilului de întrerupere.

Indicatori afectați: P, S, Z

Indicatori anulați: H, N



**132. LD I, A ED 47**

Încarcă acumulatorul în registrul vectorului de întrerupere.

Indicatori afectați: nici unul

**133. LD R, A ED 4F**

Copiază acumulatorul în registrul de reîmprospătare a memoriei.

Indicatori afectați: nici unul.

**134. LD r, (HL); LD A, (HL) 7E; LD B, (HL) 49; LD C, (HL) 4E;  
LD D, (HL) 56; LD E, (HL) 5E; LD H, (HL) 66;  
LD L, (HL) 6E;**

**135. LD r, (IX+dd); LD A, (IX+dd) DD 7Edd; LD B, (IX+dd) DD 46dd;  
LD C, (IX+dd) DD 4Edd; LD D, (IX+dd) DD 56dd;  
LD E, (IX+dd) DD 5Edd; LD H, (IX+dd) DD 66dd;  
LD L, (IX+dd) DD 6Edd;**

**136. LD r, (IY+dd); LD A, (IY+dd) FD 7Edd; LD B, (IY+dd) FD 46dd;  
LD C, (IY+dd) FD 4Edd; LD D, (IY+dd) FD 56dd;  
LD E, (IY+dd) FD 5Edd; LD H, (IY+dd) FD 66dd;  
LD L, (IY+dd) FD 6Edd;**

Încarcă registrul r cu conținutul celulei de memorie, cu adresa specificată în HL sau IX/IY+dd.

**137. LD r, r2; LD A, A 7F; LD A, B 78; LD A, C 79; LD A, D 7A;  
LD A, E 7B; LD A, H 7C; LD A, L 7D;  
LD B, A 47; LD B, B 40; LD B, C 41; LD B, D 42;  
LD B, E 43; LD B, H 44; LD B, L 45;  
LD C, A 4F; LD C, B 48; LD C, C 49; LD C, D 4A;  
LD C, E 4B; LD C, H 4C; LD C, L 4D;  
LD D, A 57; LD D, B 50; LD D, C 51; LD D, D 52;  
LD D, E 53; LD D, H 54; LD D, L 55;  
LD E, A 5F; LD E, B 58; LD E, C 59; LD E, D 5A;  
LD E, E 5B; LD E, H 5C; LD E, L 5D;  
LD H, A 67; LD H, B 60; LD H, C 61; LD H, D 62;  
LD H, E 63; LD H, H 64; LD H, L 65;  
LD L, A 6F; LD L, B 68; LD L, C 69; LD L, D 6A;  
LD L, E 6B; LD L, H 6C; LD L, L 6D;**

**138. LD r, nn; LD A, nn 3E nn; LD B, nn 06 nn; LD C, nn 0E nn;  
LD D, nn 16 nn; LD E, nn 1E nn; LD H, nn 26 nn;  
LD L, nn 2E nn;**

Încarcă registrul r cu conținutul registrului r2 sau cu constanta nn.

**139. LD A, (nnnn) 3A nnnn**

Încarcă registrul A, cu octetul din memorie a cărei adresa este dată de nnnn.

**140. LD BC, (nnnn) ED 4Bnnn**

**141. LD DE, (nnnn) ED 5Bnnnn**

Încarcă octeții mai puțin semnificativi C și E, în memorie la adresa nnnn și octeții mai semnificativi B și D, la adresa nnnn+1.

**142. LD HL, (nnnn) 2A nnnn**

Încarcă registrul L, cu conținutul celulei de memorie cu adresa nnnn și registrul H, cu conținutul celulei de memorie cu adresa nnnn+1.

**143. LD BC, nnnn 01 nnnn**



144. LD DE, nnnn      11 nnnn  
 145. LD HL, nnnn      21 nnnn  
 146. LD SP, nnnn      31 nnnn  
 147. LD IX, nnnn      DD 21nnnn  
 148. LD IY, nnnn      FD 21nnnn

Încarcă registrul dublu specificat, cu constanta de 16 biți nnnn.

149. LD IX, (nnnn)      DD 2Annnn  
 150. LD IY, (nnnn)      FD 2Annnn  
 151. LD SP, (nnnn)      ED 7Bnnnn

Încarcă octeții inferioari ai registrelor IX, IY, SP, din memorie, de la adresa nnnn și octeții superiori, de la adresa nnnn+1

152. LD SP, HL          F9  
 153. LD SP, IX          DD F9  
 154. LD SP, IY          FD F9

Încarcă registrul indicator de stivă SP, cu conținutul registrului, de 16 biți, specificat.

155. LDD              ED A8  
 156. LDDR            ED B8  
 157. LDI              ED A0  
 158. LDIR            ED B0

Încarcă octetul din memorie, de la locația cu adresa indicată de HL, în locația cu adresa indicată de DE. Decrementează contorul de 16 biți BC. Incrementează/decrementează după încărcare HL și DE, în cazul instrucțiunilor LDL, LDIR/LDD, LDDR. Repetă operația, pentru LDDR și LDIR, până când BC a fost decrementat la zero.

159. NEG            ED 44

Instrucțiunea realizează complementul față de doi al acumulatorului.

Indicatori afectați: toți.

160. NOP            00

UCP nu efectuează nici o operație.

Indicatori afectați: nici unul.

161. OR (HL)          B6  
 162. OR (IX+dd)      DD B6dd  
 163. OR (IY+dd)      FD B6dd

Adună logic conținutul celulei de memorie, cu adresa specificată în HL sau IX/IY+dd, la acumulator.

Indicatori afectați: P, S, Z

Indicatori anulați: C, H, N

164. OR r;            OR A B7; OR B B0; OR C B1; OR D B2;  
                          OR E B3; OR H B4; OR L B5;

Adună logic conținutul registrului r la acumulator. Instrucțiunea OR A este utilă pentru a testa indicatorii de paritate, semn și zero, deoarece nu modifică conținutul lui A.

Indicatori afectați: P, S, Z



**Indicatori anulați: C, H, N**

**165. OR nn F6 nn**

Adună logic octetul **nn** la acumulator. Instrucțiunea se poate folosi pentru a poziționa în unu biți individuali din **A**. De exemplu, **OR 40H** va poziționa bitul 6, al acumulatorului, în unu.

**Indicatori afectați: P, S, Z**

**Indicatori anulați: C, H, N**

**166. OTDR ED BB**

**167. OTIR ED B3**

Transferă la portul de ieșire, cu adresa dată în registrul **C**, conținutul locației de memorie cu adresa specificată de **HL**. Registrul **B** este decrementat. Perechea de registre **HL** este incrementată/decrementată la execuția instrucțiunilor **OTIR/OTDR**. Procesul se repetă pînă cînd **B** devine zero.

**Indicatori afectați: N, Z.**

**168. OUT (C), r OUT (C), A ED 79; OUT (C), B ED 41;  
OUT (C), C ED 49; OUT (C), D ED 51;  
OUT (C), E ED 59; OUT (C), H ED 61;  
OUT (C), L ED 69;**

Transferă octetul din registrul **r**, la portul de ieșire cu adresa specificată în registrul **C**.

**Indicatori afectați: nici unul.**

**169. OUT (nn), A D3 nn**

Transferă octetul din acumulator, la portul de ieșire cu adresa **nn**.

**Indicatori afectați: nici unul.**

**170. OUTD ED AB**

**171. OUTI ED A3**

Transferă octetul din locația de memorie cu adresa specificată în **HL**, la portul de ieșire cu adresa dată în registrul **C**. Registrul **B** este incrementat. Perechea de registre **HL** este incrementată/decrementată de instrucțiunile **OUTI/OUTD**.

**Indicatorii afectați: Z**

**Indicatori poziționați în unu: N**

**172. POP AF F1**

**173. POP BC C1**

**174. POP DE D1**

**175. POPHL E1**

**176. POPIX DD E1**

**177. POP IY FD E1**

Încarcă perechea de registre sau registrele duble specificate, cu conținutul primelor două celule din virful stivei. Octetul din virful stivei (specificat de **SP** se va încărca în octetul mai puțin semnificativ al perechii/registrului dublu după care **SP** se va incrementa și se va încărca conținutul celei de-a doua celule din stivă, în octetul cel mai semnificativ al perechii de registre/registrului dublu. În final indicatorul de stivă **SP** se va mai incrementa o dată.

**Indicatori afectați: toți — pentru POP AF și nici unul — pentru celelalte instrucțiuni.**

**178. PUSH AF F5**

**179. PUSH BC C5**

**180. PUSH DE D5**



**181. PUSH HL E5**

**182. PUSH IX DD E5**

**183. PUSH IY FD E5**

Plasează în stivă conținutul perechii de registre sau al registrului dublu. Mai întâi se decrementează SP și se memorează octetul mai semnificativ, apoi se decrementează din nou SP și se memorează octetul mai semnificativ, al perechii/registrului dublu.

Indicatori afectați: nici unul.

**184. RES b, (HL);** RES 0, (HL) CB 86; RES 1, (HL) CB 8E;  
RES 2, (HL) CB 96; RES 3, (HL) CB 9E;  
RES 4, (HL) CB A6; RES 5, (HL) CB AE;  
RES 6, (HL) CB B6; RES 7, (HL) CB BE;

**185. RES b, (IX+dd);** RES 0, (IX+dd) DD CBdd86;  
RES 1, (IX+dd) DD CBdd8E;  
RES 2, (IX+dd) DD CBdd96;  
RES 3, (IX+dd) DD CBdd9E;  
RES 4, (IX+dd) DD CBddA6;  
RES 5, (IX+dd) DD CBddAE;  
RES 6, (IX+dd) DD CBddE6;  
RES 7, (IX+dd) DD CBddB6;

**187. RES b, (IY+dd);** RES 0, (IY+dd) FD CBdd86;  
RES 1, (IY+dd) FD CBdd8E;  
RES 2, (IY+dd) FD CBdd96;  
RES 3, (IY+dd) FD CBdd9E;  
RES 4, (IY+dd) FD CBddA6;  
RES 5, (IY+dd) FD CBddAE;  
RES 6, (IY+dd) FD CBddE6;  
RES 7, (IY+dd) FD CBddB6;

**188. RES b, r;** RES 0,A CB 87; RES 0,B CB 80; RES 0,1 CB 81;  
RES 1,A CB 8F; RES 1,B CB 88; RES 1,C CB 89;  
RES 2,A CB 97; RES 2,B CB 90; RES 2,C CB 91;  
RES 3,A CB 9F; RES 3,B CB 98; RES 3,C CB 99;  
RES 4,A CB A7; RES 4,B CB A0; RES 4,C CB A1;  
RES 5,A CB AF; RES 5,B CB A8; RES 5,C CB A9;  
RES 6,A CB B7; RES 6,B CB B0; RES 6,C CB B1;  
RES 7,A CB BF; RES 7,B CB B8; RES 7,C CB B9;  
RES 0,D CB 82; RES 0,E CB 83; RES 0,H CB 84;  
RES 1,D CB 8A; RES 1,E CB 8B; RES 1,H CB 8C;  
RES 2,D CB 92; RES 2,E CB 93; RES 2,H CB 94;  
RES 3,D CB 9A; RES 3,E CB 9B; RES 3,H CB 9C;  
RES 4,D CB A2; RES 4,E CB A3; RES 4,H CB A4;  
RES 5,D CB AA; RES 5,E CB AB; RES 5,H CB AC;  
RES 6,D CB B2; RES 6,E CB B3; RES 6,H CB B4;  
RES 7,D CB BA; RES 7,E CB BB; RES 7,H CB BC;  
RES 0,L CB 85;  
RES 1,L CB 8D;  
RES 2,L CB 95;  
RES 3,L CB 9D;



RES 4,L CB A5;  
 RES 5,L CB AD;  
 RES 6,L CB B5;  
 RES 7,L CB BD;

Anulează bitul  $b$ , din celula de memorie, cu adresa specificată de HL sau IX/IY+dd, sau bitul  $b$ , al registrului r.

189. RET C9  
 190. RET C D8  
 191. RET M F8  
 192. RET NC D0  
 193. RET NZ CO  
 194. RET P FO  
 195. RET PE E8  
 196. RET PO E0  
 197. RET Z C8

Revenire necondiționată/condiționată, din subrutină. Dacă condiția este îndeplinită, conținutul primelor două celule din vârful stivei se transferă în PC. SP este incrementat de două ori.

198. RETI ED 4D  
 199. RETN ED 45

Revenire din întrerupere mascabilă (RETI) nemascabilă (RETN).

Următoarele instrucțiuni RL și RLA rotesc biții la stînga, incluzînd și bitul de transport. Se considera indicatorul de transport C, concatenat la stînga cuvîntului cu biți  $b_7b_6b_5b_4b_3b_2b_1b_0$ ,

$C, b_7b_6b_5b_4b_3b_2b_1b_0 \leftarrow b_7b_6b_5b_4b_3b_2b_1b_0, C$

200. RL (HL) CB 16  
 201. RL (IX+dd) DD CBdd16  
 202. RL (IY+dd) FD CBdd16  
 203. RL r RL A CB 17; RL B CB 10; RL C CB 11;  
 RL D CB 12; RL E CB 13; RL F CB 14;  
 RL H CB 15;

Octetul din memorie, cu adresa specificată în HL sau IX/IY+dd, sau conținutul registrului r sînt rotite la stînga, prin concatenarea cu indicatorul de transport.

Indicatori afectați: C, P, S, Z.

Indicatori anulați: H, N

204. RLA 17

Conținutul acumulatorului, concatenat cu indicatorul de transport C, este rotit la stînga cu un bit. RLA efectuează aceeași operație ca și RL A, însă de două ori mai repede.

Indicatori afectați: C

Indicatori anulați: H, N

Următoarele instrucțiuni RLC și RLCA rotesc biții spre stînga, fără concatenarea indicatorului de transport C. Indicatorul de transport se încarcă cu conținutul bitului cel mai semnificativ, al cuvîntului care este rotit.

$b_7b_6b_5b_4b_3b_2b_1b_0 \leftarrow b_6b_5b_4b_3b_2b_1b_0b_7; C \leftarrow b_7$



205. RLC (HL) CB 066

206. RLC (IX+dd) DD CBdd06

207. RLC (IY+dd) FD CBdd06

208. RLC r RLC A CB 07; RLC B CB 00; RLC C CB 01;  
RLC D CB 02; RLC E CB 03; RLC H CB 04;  
RLC L CB 05;

Conținutul octetului din memorie, cu adresa specificată în HL sau IX/IY + +dd, sau conținutul registrului r se rotește la stînga. Bitul 7 se deplasează în bitul 0 și în indicatorul C.

Indicatori afectați: C, P, S, Z

Indicatori anulați: H, N.

209. RLCA 07

Conținutul acumulatorului este rotit cu un bit spre stînga. Bitul 7 al acumulatorului se transferă în bitul 0 și în indicatorul de transport C. RLCA efectuează aceeași operație ca și RLC A, dar de două ori mai repede.

Indicatori afectați: C

Indicatori anulați: H, N

210. RLD ED 6F

Rotire a 4 biți pe o distanță de 12 biți: ultimii 4 biți din A se deplasează în ultimii 4 biți din celula de memorie specificată de HL; ultimii 4 biți din celula de memorie se deplasează în primii 4 biți mai semnificativi ai celulei; în timp ce primii 4 biți mai semnificativi ai celulei se deplasează în ultimii 4 biți ai acumulatorului. Primii 4 biți ai acumulatorului rămîn neschimbați.

$A_{7:4}, A_{3:0}, b_{7:4}, b_{3:0} \leftarrow A_{7:4}, b_{7:4}, b_{3:0}, A_{3:0}$

Indicatori afectați: P, S, Z

Indicatori anulați: H, N

Următoarele instrucțiuni RR și RRA rotesc spre dreapta conținutul cuvîntului specificat, concatenat la dreapta cu indicatorul de transport C.

211. RR (HL) CB 1E

212. RR (IX+dd) DD CBdd1E

213. RR (IY+dd) FD CBdd1E

214. RR r RR A CB 1F; RR B CB 18; RR C CB 19;  
RR D CB 1A; RR E CB 1B; RR H CB 1C;  
RR L CB 1D;

Conținutul cuvîntului din memorie, a cărui adresă este dată de HL sau IX/IY+dd, sau registrul r, concatenat la dreapta cu indicatorul C, se rotesc la dreapta cu un bit;  $b_7$  se transferă în C, C se transferă în  $b_0$ .

$b_7b_6b_5b_4b_3b_2b_1b_0, C \leftarrow C, b_7b_6b_5b_4b_3b_2b_1b_0$

Indicatori afectați: C, P, S, Z

Indicatori anulați: H, N

215. RRA 1F

Rotește spre dreapta conținutul acumulatorului, concatenat la dreapta cu indicatorul C. RRA execută aceeași operație ca și RRA, dar de două ori mai repede.

Indicatori afectați: C

Indicatori anulați: H, N

Următoarele instrucțiuni RRC și RRCA rotesc biții cuvîntului la dreapta. Bitul 0 este forțat, atît în bitul 7, cît și în indicatorul C, plasat la dreapta cuvîntului specificat.



216. RRC (HL) CB OE  
 217. RRC (IX+dd) DD CBddOE  
 218. RRC (IX+dd) FD CBddOE  
 219. RRC r RRC A CB OF; RRC B CB 08; RRC C CB 09;  
 RRC D CB 0A; RRC E CB 0B; RRC F CB 0C;  
 RRC H CB 0D;

Conținutul cuvintului din memorie, cu adresa specificată în HL sau IX/IX+dd, sau conținutul registrului r se rotește la dreapta. Bitul  $b_0$  este transferat în  $b_7$  și în C.

$b_7b_6b_5b_4b_3b_2b_1b_0, C \leftarrow b_0b_7b_6b_5b_4b_3b_2b_1b_0$

Indicatori afectați: C, P, S, Z

Indicatori anulați: H, N

## 220. RRCA OF

Execută aceeași operație ca și RRCA, dar de două ori mai repede.

## 221. RRD ED 67

Rotire a 4 biți, pe distanța de 12 biți: ultimii 4 biți ai acumulatorului se transferă în primii 4 biți ai celulei de memorie specificată de HL; primii 4 biți ai celulei se deplasează în ultimii 4 biți ai celulei, în timp ce ultimii 4 biți ai celulei se transferă în ultimii 4 biți ai acumulatorului.

$A_{7:4}, A_{3:0}, b_{7:4}, b_{3:0} \leftarrow A_{7:4}, b_{3:0}, A_{3:0}, b_{7:4}$

Instrucțiunea este folosită pentru operații în binar-zecimal.

Indicatori afectați: P, S, Z

Indicatori anulați: H, N

## 222. RST 00H C7

## 223. RST 08H CF

## 224. RST 10H D7

## 225. RST 18H DF

## 226. RST 20H EF

## 227. RST 28H FF

## 228. RST 30H F7

## 229. RST 38H FF

Instrucțiuni de restart, pe un octet, care asigură chemări de subrutine, de la adresele specificate în hexa.

## 230. SBC A, (HL) 9E

## 231. SBC A, (IX/IX+dd) DD 9Edd

## 232. SBC A, (IX/IX+dd) FD 9Edd

## 233. SBC A, r SBC A,A 9F; SBC A,B 98; SBC A,C 99; SBC A,D 9A; SBC A,E 9B; SBC A,H 9C; SBC A,L 9D;

Scade conținutul celulei de memorie, cu adresa specificată în HL sau IX/IX+dd, sau conținutul registrului r, și conținutul indicatorului de transport C, din acumulator.

Indicatori afectați: C, H, O, S, Z

Indicatori poziționați în unu: N

## 234. SBC A,nn DE nn

Scade octetul imediat nn și conținutul indicatorului de transport C, din acumulator.

Indicatori poziționați: C, H, O, S, Z

Indicatori poziționați în unu: N

## 265. SBC HI, BC ED 42



236. SBC HL, DE ED 52  
 237. SBC HL, HL ED 62  
 238. SBC HL, SP ED 72

Scade registrul dublu specificat și indicatorul C, din perechea HL, cu rezultatul în HL. Înainte de a folosi aceste instrucțiuni se impune, în unele cazuri, anularea indicatorului C, prin efectuarea instrucțiunii OR A.

Indicatori afectați: C, H, O, S, Z

Indicatori poziționați în unu: N

### 239. SCF 37

Poziționează în unu indicatorul C.

Indicatori afectați: C

Indicatori anulați: H, N

240. SET b, (HL) SET 0, (HL) CB C6; SET 1, (HL) CB CE;  
 SET 2, (HL) CB D6; SET 3, (HL) CB DE;  
 SET 4, (HL) CB E6; SET 5, (HL) CB EE;  
 SET 6, (HL) CB F6; SET 7, (HL) CB FE;

241. SET b, (IX+dd) SET 0, (IX+dd) DD CBddC6;  
 SET 1, (IX+dd) DD CBddCE;  
 SET 2, (IX+dd) DD CBddD6;  
 SET 3, (IX+dd) DD CBddDE;  
 SET 4, (IX+dd) DD CBddE6;  
 SET 5, (IX+dd) DD CBddEE;  
 SET 6, (IX+dd) DD CBddF6;  
 SET 7, (IX+dd) DD CBddFE;

242. SET b, (IY+dd) SET 0, (IY+dd) FD CBddC6;  
 SET 1, (IY+dd) FD CBddCE;  
 SET 2, (IY+dd) FD CBddD6;  
 SET 3, (IY+dd) FD CBddDE;  
 SET 4, (IY+dd) FD CBddE6;  
 SET 5, (IY+dd) FD CBddEE;  
 SET 6, (IY+dd) FD CBddF6;  
 SET 7, (IY+dd) FD CBddFE;

243. SET b, r SET 0,A CB C7; SET 0,B CB C0; SET 0,C CB C1;  
 SET 1,A CB CF; SET 1,B CB C8; SET 1,C CB C9;  
 SET 2,A CB D7; SET 2,B CB D0; SET 2,C CB D1;  
 SET 3,A CB DF; SET 3,B CB D8; SET 3,C CB D9;  
 SET 4,A CB E7; SET 4,B CB E0; SET 4,C CB E1;  
 SET 5,A CB EF; SET 5,B CB E8; SET 5,C CB E9;  
 SET 6,A CB F7; SET 6,B CB F0; SET 6,C CB F1;  
 SET 7,A CB FF; SET 7,B CB F8; SET 7,C CB F9;  
 SET 0,D CB C2; SET 0,E CB C3; SET 0,H CB C4;  
 SET 1,D CB C9; SET 1,E CB CA; SET 1,H CB CB;  
 SET 2,D CB D2; SET 2,E CB D3; SET 2,H CB D4;  
 SET 3,D CB DA; SET 3,E CB DB; SET 3,H CB DC;  
 SET 4,D CB E2; SET 4,E CB E3; SET 4,H CB E4;  
 SET 5,D CB EA; SET 5,E CB EB; SET 5,H CB EC;  
 SET 6,D CB F2; SET 6,E CB F3; SET 6,H CB F4;  
 SET 7,D CB FA; SET 7,E CB FB; SET 7,H CB FC;  
 SET 0,L CB C5;  
 SET 1,L CB CD;



SET 2,L CB D5;  
 SET 3,L CB DD;  
 SET 4,L CB E5;  
 SET 5,L CB ED;  
 SET 6,L CB F5;  
 SET 7,L CB FD;

Se poziționează în unu bitul b, al cuvîntului de memorie, cu adresa specificată în HL sau IX/IY+dd, sau al registrului r.

Indicatori afectați: nici unul

Indicatori anulați: N.

Următoarele instrucțiuni SLA deplasează biții spre stînga.

244. SLA (HL)

245. SLA (IX+dd) DD CBdd26

246. SLA (IY+dd) FD Cbdd26

247. SLA r SLA A CB 27; SLA B CB 20; SLA C CB 21;  
 SLA D CB 22; SLA E CB 23; SLA H CB 24;  
 SLA L CB 25;

Deplasează aritmetic, spre stînga, conținutul octetului de memorie, cu adresa specificată în HL sau IX/IY+dd sau conținutul registrului r. Bitul 7 se forțează în indicatorul de transport C. În bitul 0 se forțează 0. Această operație dublează conținutul cuvîntului. Instrucțiunea SLA A efectuează aceeași operație ca și ADDA,A dar de două ori mai repede.

Indicatori afectați: C, P, S, Z

Indicatori anulați: H, N.

Instrucțiunile SRA deplasează aritmetic biții cuvîntului spre dreapta.

248. SRA (HL) CB 2E

249. SRA (IX+dd) DD CBdd2E

250. SRA (IY+dd) FD CBdd2E

251. SRA r SRA A CB 2F; SRA B CB 28; SRA C CB 29;  
 SRA D CB 2A; SRA E CB 2B; SRA H CB 2C;  
 SRA L CB 3E;

Deplasează aritmetic, spre dreapta, conținutul celulei de memorie, cu adresa specificată în HL sau IX/IY+dd, sau conținutul registrului r. Bitul 7 se conservă în poziția 7 și se extinde, de asemenea, în poziția 6; bitul 0 se transferă în indicatorul C. Indicatorul C va fi poziționat în unu, dacă numărul original a fost impar.

Indicatori poziționați: C, P, S, Z

Următoarele instrucțiuni SRL deplasează logic informația spre dreapta, cu forțarea bitului 7, în zero.

252. SRL (HL) CB 3E

253. SRL (IX+dd) DD CBdd3E

254. SRL (IY+dd) FD CBdd3E

255. SRL r SRL A CB 3F; SRL B CB 38; SRL C CB 39;  
 SRL D CB 3A; SRL E CB 3B; SRL H CB 3C;  
 SRL L CB 3D;

Deplasează logic spre dreapta conținutul cuvîntului din memorie, cu adresa specificată în HL sau IX/IY+dd, sau conținutul registrului r. În bitul 7 se forțează 0, iar în indicatorul de transport C, se forțează conținutul bitului 0.

Indicatori afectați: C, P, Z



Indicatori anulați: H, N, S

256. SUB (HL) 96

257. SUB (IY+dd) FD 96 DD 96dd

258. SUB (IY+dd) FD 96dd

259. SUB r SUB A 97; SUBO B 90; SUB C 91; SUB D 92;  
SUB E 93; SUB H 94; SUB L 95;

Scade conținutul celulei de memorie, a cărei adresă este specificată de HL sau IX/IY+dd, sau conținutul registrului r, din acumulator.

Indicatori, afectați: C, H, O, S, Z

Indicatori anulați: N

260. SUB nn

Scade din acumulator operandul imediat nn.

Indicatori afectați: C, H, O, S, Z

Indicatori poziționați în unu: N

261. XOR (HL) AE

262. XOR (IX+dd) DD AEdd

263. XOR (IY+dd) FD AEdd

264. XOR r XOR A AF; XOR B A8; XOR C A9; XOR D AA;  
XOR E AB; XOR H AC; XOR L AD;

Adună logic conținutul celulei de memorie, cu adresa specificată în HL sau IX/IY+dd, sau conținutul registrului r, la acumulator.

Indicatori afectați: P, S, Z

Indicatori anulați: C, H, N

265. XOR nn E nn

Adună logic operandul imediat nn, la conținutul acumulatorului.

Indicatori afectați: P, S, Z

Indicatori anulați: C, H, N.



## Încărcarea și lansarea în BASIC a unor programe scrise în limbaj de asamblare cod mașină

Întrucît microcalculatorul HC-85 este realizat pe baza microprocesorului Z80, instrucțiunile prezentate în capitolul precedent pot fi utilizate pentru scrierea unor programe eficiente sub aspectul vitezei de operare și al spațiului ocupat în memorie.

Utilizarea instrucțiunilor microprocesorului Z80 se poate face în două moduri diferite.

Astfel, una din soluții constă în citirea de pe casetă a unui program numit *asambler*, cu ajutorul căruia, un program sursă, scris în limbaj de asamblare, folosind mnemonicele date pentru instrucțiuni, se va transforma într-un program obiect, în cod binar, direct executabil.

Această metodă permite scrierea unor programe de dimensiuni suficient de mari, pentru diferite aplicații cu caracter profesional. În acest sens se recomandă folosirea pachetelor de programe GENS 3 și MONS 3. Ele pot fi folosite pentru editarea și asamblarea de programe, cit și pentru dezasamblarea unor rutine date în cod mașină.

Cea de-a doua metodă se bazează pe scrierea programelor direct în cod mașină și încărcarea lor, respectiv — lansarea în execuție sub controlul unui program scris în BASIC. Deoarece HC-85 are interpretorul pentru limbajul BASIC stocat în memorie (fiind activat imediat după aplicarea tensiunii de alimentare), în cele ce urmează se vor examina posibilitățile oferite de cea de-a doua metodă.

Facilitățile limbajului BASIC HC-85 permit apelarea ca subrutine, a unor programe scrise în cod mașină.

Pentru implementarea programelor în cod mașină, în continuare, se vor face unele precizări.

Inițial este bine ca problemele să fie scrise în limbaj de asamblare, pornind de la materialul referitor la instrucțiunile microprocesorului Z80, date anterior.

Instrucțiunile limbajului de asamblare vor fi înlocuite cu echivalentele lor hexazecimale.

Deoarece programul în cod mașină va fi introdus cu instrucțiunea DATA, pentru BASIC HC-85 se impune conversia codurilor hexazecimale, în coduri zecimale. În acest scop se vor avea în vedere cîmpurile instrucțiunii: codul de operație, operandul pe 8/16 biți, adresa de 16 biți, și modul lor de încărcare în memorie, sub forma de cuvinte, reprezentînd numere zecimale, cuprinse în gama 0—255, memoria fiind organizată la nivel de octet.

Exemple:

Încărcarea registrului A cu constanta 14=0EH

Limbaj asamblare;

LD A, 0EH

Cod mașină hexa;

3E 0E

Cod mașină zecimal

62 14



Încărcarea registrului B cu constanta 255=FF

LD B, FFH

06 FF

06 255

Încărcarea registrului dublu HL cu constanta 7F FF 32767.

Încărcarea adreselor date în baza zece, în memorie, necesită reprezentarea lor sub forma conținutului zecimal, a două cuvinte succesive de memorie. În acest sens numărul 32767 va fi scris sub forma următoare: 127 255, deoarece se poate observa că  $127 * 256 = 32512$ , iar  $32512 + 255 = 32767$

Se reamintește că în reprezentarea unui cuvint de 16 biți într-o instrucțiune, mai întâi va fi scris octetul mai puțin semnificativ, după care octetul mai semnificativ; FF și — respectiv 7F, în exemplul de față.

Încărcarea registrului dublu HL, cu constanta 7F FF se va realiza prin instrucțiunile următoare:

LD HL, 7FFFFH

21 7F FF

33 255 127

**Exemplu:**

Să se scrie un program în limbaj de asamblare, care va compara două numere pozitive, aflate în două celule consecutive de memorie, stocînd cel mai mare număr în cea de-a treia celulă consecutivă.

Cele trei celule de memorie vor avea următoarele adrese:

FD4F 32015

FD10 32016

FD11 32017

Se consideră că cele două numere care se compară sînt deja introduse la adresele 32015 și 32016.

Limbaj asamblare;	Cod hexazecimal;	Cod zecimal
LD HL, 0FFDH	21 0F FD	33 15 125
LD A, (HL)	7E	126
INC A, HL	23	35
CP (HL)	BE	190
JR NC, ETC	30 01	48 1
LD A, (HL)	7E	126
ETC: INC HL	77	119
RET	C9	201

Pentru a introduce în memoria calculatorului HC-85 acest program și pentru a-l executa, se poate folosi următorul program BASIC (se consideră că programul în cod mașină se introduce în memorie, începînd cu adresa 32001).

10 CLEAR 32000

20 RESTORE

30 FOR a=32001 TO 32012:READ X

40 POKE a,x:NEXT a

50 DATA 33, 15, 125, 126, 35, 190, 48, 1, 126, 35, 119, 201

60 INPUT "x"; x : INPUT "y="; y

70 POKE 32015,x : POKE 32016,y

80 CLS : RANDOMIZE URS 32001



## 90 FOR a=32015 TO 32017 : PRINT PEEK a : NEXT a

Instrucțiunea **10** anulează conținutul celulelor de memorie începînd cu adresa **32000**, pînă la adresa **65535**.

Instrucțiunea **20** reinițializează indicatorul la prima instrucțiune **DATA**.

Instrucțiunile **30**, **40**, **50** asigură încărcarea în memorie a programului în cod mașină.

Instrucțiunile **60** și **70** citesc de la tastatură cele două numere, care se compară, și le introduc în celulele de memorie, cu adresele: **32015** și **32016**.

Instrucțiunea **80** șterge ecranul și lansează în execuție rutina în cod mașină, de la adresa **32001**, cu instrucțiunea **RANDOMIZE USR 32001** (în locul ei se poate folosi instrucțiunea **PRINT USR 32001**).

Programul în cod mașină se termină cu instrucțiunea **RET** (revenire din subrutină). După execuția acestui program, controlul este returnat programului **BASIC**, la instrucțiunea **90**.

Instrucțiunea **90** permite afișarea pe ecran a conținutului celor trei celule de memorie.

### Exemplu:

Să se scrie un program în limbaj de asamblare, care va permite generarea la difuzor a unui sunet specific. *Se va folosi rutina BEEP, stocată în EPROM la adresa 949.*

După cum este cunoscut, controlul înălțimii și duratei sunetelor, generate prin programe **BASIC**, este destul de limitat ca viteză. Folosirea codului mașină va permite accelerarea procesului, obținîndu-se efecte sonore remarcabile în cadrul programelor, care cheama asemenea rutine.

Sucesiunea de operații este următoarea:

1. inițializare înălțime, inițializare durată,
2. chemare rutină de sunet din EPROM,
3. restaurare parametri sunet,
4. modificare înălțime sunet,
5. dacă înălțimea sunetului nu a atins valoarea limită se trece la punctul 2,
6. sfîrșit program sunet.

Pe scurt programul în cod mașină va repeta o notă pentru care înălțimea și durata au fost stocate în **HL** și **DE**, reducînd înălțimea pînă la atingerea valorii limită. Procesul se repetă de un număr dat de ori.

Limbaaj de asamblare,	Cod hexa,	Cod zecimal,	Comentarii.
LDB 10	06 0A	6, 10	;Încarcă contorul ;B de repetare.
CONT: PUSH BC	C5	197	;Salvează contorul ;în stivă
LD HL,0	21 00 00	33, 0, 0	;Valoare inițială ;Înălțime
BUCLA: LD DE,100	11 64 00	17, 100, 0	;Durata unui su- ;net.
PUSH HL	E5	229	;Salvează valoarea ;înălțimii.
CAL 949	CD B5 03	205, 181, 3	;Cheamă rutina ;BEEP.



LD BC,20	01 14 00	1,20,0	;Încarcă valoarea ;pasului de modifi-
LD DE 100	11 64 00	17,100,0	;care a înălțimii.
POP HL	E1	225	;Încarcă valoarea ;limită a sunetului
ADD A,0	C6 00	198,0	;Recuperează valo-
ADC HL,BC	ED 4A	237,74	;rea înălțimii din
PUSH HL	E5	229	;stivă.
ADD A,0	C6 00	198,0	;Anulează bitul de
SBC HL,DE	ED 52	237,82	;transport.
POP HL	E1	225	;Noua valoare a
JR C,BUCLA	38 E6	56,230	;înălțimii.
(JR C,-26)			;Salvează în stivă
POP BC	C1	193	;valoarea înălțimii
DJ NZ CONT	10 DF	16,223	;Anulează bitul de
(DJ NZ,-33)			;transport.
RET	C9	201	;Testează valoarea
			;limită a înălțimii
			;Recuperează din
			;stivă valoarea
			;înălțimii.
			;S-a atins limita
			;înălțimii
			;Recuperează va-
			;loarea contorului
			;de repetare.
			;Se repetă?
			;Revenire în ru-
			;tină.

Programul are 36 de octeți și va fi încărcat de la adresa 65332, până la adresa 65367, în RAM.

1 REM PROGRAM SIRENA

10 CLEAR 65331

20 RESTORE

30 FOR a=65332 TO 65367 : READ x

40 POKE a,x : NEXT a

50 DATA 6, 10, 197, 33, 0, 0, 17, 100, 0, 229,

205, 181, 3, 1, 20, 0, 17, 100, 0, 225,

198, 0, 237, 74, 229, 198, 0, 237, 82

225, 56, 230, 193, 16, 223, 201

60 RANDOMIZE USR 65332



## Capitolul 25. | Calculatorul personal profesional Felix PC și familia IBM-PS/2

### 25.1. Calculatorul personal – profesional FELIX PC\*

Sistemul FELIX PC este un calculator realizat în țara noastră, cu un grad de integrare tehnologică ridicat, structură compactă și un sistem de programe ce acoperă o gamă largă de aplicații.

FELIX PC are o structură modulară, fiind alcătuit dintr-un modul de bază și module de extensie. Modulul de bază cuprinde resursele hardware care asigură funcționarea sa ca sistem universal, cu configurație redusă care include: unitatea centrală, tastatura, consola serială, imprimanta, discuri flexibile. Modulele de extensie sînt opționale, fiind utilizate în alcătuirea unor configurații adecvate aplicațiilor sau pentru mărirea disponibilității și facilităților sistemului.

#### Structura modului de bază

Modulul de bază conține următoarele resurse:

- unitatea de prelucrare bazată pe microprocesorul 8088/8086 și coprocesorul 8087;
- memorie RAM de 256 Kocteți;
- memorie EPROM de 12 Kocteți — 96 Kocteți;
- cuplor pentru discuri flexibile de 5 1/4" sau 8";
- interfețe pentru:
  - tastatură;
  - imprimantă serială;
  - comunicație asincronă/sincronă;
  - generator de tonuri;
- ceas de timp real;
- numărătoare programabile;
- sistem de întreruperi;

---

\* Proiectat la Institutul Politehnic București (prof. dr. ing. Adrian Petrescu, șl. dr. ing. Trandafir Moisa, șl. dr. ing. Nicolae Tăpuș, șl. dr. ing. Irina Athanasu) în colaborare cu Întreprinderea de Calculatoare electronice (ing. Florea Tănase, ing. Traian Mihu, ing. Tudorel Domocoș, ing. Gabriel Drăghicescu și ing. Sandu Anghel).



- canale de acces direct la memorie;
- conectori pentru module de extensie;
- conectori pentru periferice.

Sistemul FELIX-PC are o magistrală de date configurabilă pe 16 biți sau 8 biți, în funcție de microprocesorul care se utilizează: 8086 sau 8088.

Cele două microprocesoare sînt compatibile la nivel de pini și la nivel de set de instrucțiuni, numai că microprocesorul 8088 comunică cu exteriorul printr-o magistrală de date de 8 biți, iar 8086 comunică cu exteriorul printr-o magistrală de date de 16 biți (8088 execută operațiile de transfer cu exteriorul pe cuvint în două cicluri mașină, iar 8086 într-un singur ciclu).

Microprocesoarele pot să adreseze 1 Moctet de memorie, avînd în afară de instrucțiunile comune altor microprocesoare și instrucțiuni de lucru pe șiruri de caractere, instrucțiuni de înmulțire și împărțire în virgulă fixă cu diverse moduri de adresare.

Frecvența de lucru este de 5MHZ iar un ciclu mașină este de 800ns (ciclu de I/E de 1  $\mu$ s)

Microprocesorul este utilizat în modul maxim pentru a putea fi cuplat cu coprocesorul matematic 8087.

Sistemul FELIX-PC a fost prevăzut cu posibilitatea introducerii microprocesorului matematic NDP 8087. Acesta crește considerabil viteza de lucru (este de circa 100 ori mai rapid decît 8086/8088) în cazul prelucrărilor numerice care cuprind:

- adunarea în virgulă fixă și mobilă;
- scăderea în virgulă fixă și mobilă;
- înmulțirea în virgulă fixă și mobilă;
- împărțirea în virgulă fixă și mobilă;
- rădăcină pătrată;
- rotunjire;
- funcții transcendente ca: tangenta, arctangenta (parțiale), logaritm, ridicare la putere.

Reprezentarea datelor în virgulă mobilă este conform standardului IEEE.

Din punct de vedere software, nu toate translatoarele utilizează în mod implicit coprocesorul 8087 și din acest motiv utilizatorul trebuie să-și realizeze procedurile corespunzătoare pentru apelarea funcțiilor matematice. Versiunile mai noi de interpretoare și translatoare printr-o configurare corespunzătoare pot lucra implicit cu coprocesorul 8087.

Memoria RAM este realizată cu componente de memorie dinamică 4164 (64K $\times$ 1) și are o capacitate de 256 Kocteți. Este organizată pe 8 sau 16 biți, fiind prevăzută cu bit de paritate la nivel de octet.

Memoria EPROM, de capacitate între 12 Kocteți și 96 Kocteți, în funcție de tipul de componente care se utilizează, conține BIOS-ul care este format din:

- testul resurselor hardware standard;
- rutinele de I/E asociate echipamentelor standard;
- generatorul de caractere pentru modul grafic;
- încărcătorul pentru discul flexibil;
- programe pentru depanare.

În configurația standard, cuplorul pentru discuri flexibile este configurat pentru discuri de 5 1/4".

Discurile flexibile sînt sectorizate software. Cuplorul de disc permite lucrul cu discuri dublă densitate, simplă sau dublă față și utilizează codificarea MFM



(modulare în frecvență modificată). Discul flexibil este organizat pe 40 piste, cu 9 sectoare de 512 octeți pe fiecare pistă. Deci capacitatea totală a unui astfel de disc este de 184.320 de octeți pentru discurile simplă față, respectiv 368.640 de octeți pentru discurile dublă față. Rata de transfer este de 250 Kbiți pe secundă.

Configurația de bază conține și o interfață paralelă programabilă prin intermediul căreia se cuplează:

- interfața de tastatură care asigură cuplarea serială a tastaturii. Tastatura generează un număr de  $83 \times 2$  coduri, denumite coduri de scanare. Pentru fiecare tastă se generează un cod la apăsare și unul la ridicare. În acest mod se poate determina durata de acționare a unei taste. Tastatura este prevăzută cu un număr mare de taste funcționale.

- comutatoarele de configurare sistem care specifică: capacitatea memoriei RAM, tipul terminalului grafic utilizat, modul de lucru al terminalului grafic, numărul de unități de disc atașate sistemului.

- difuzorul, al cărui control se realizează în două moduri:

- direct prin program prin poziționarea unui bit de date, generându-se un tren de impulsuri;

- prin programarea canalului 2 al contorului programabil, generându-se frecvența dorită.

Ambele moduri pot să fie controlate simultan. Difuzorul este utilizat pentru semnalizarea erorilor în etapa de testare și pentru generarea de semnale audio.

- interfața de casetă magnetică audio care asigură o rată de transfer de 1 Kbauds—2 Kbauds. Transferul de date cu caseta generează și verifică CRC.

Sistemul este prevăzut cu 6 contoare programabile utilizate după cum urmează:

- canalul 0 — generează semnale periodice care asigură cuanta de bază pentru ceasul de timp real al sistemului;
- canalul 1 — generează cereri de cicluri DMA pentru operația de reîmprospătare a memoriei dinamice RAM;
- canalul 2 — generator de tonuri pentru difuzor;
- canalele 3, 4 — generează semnale de tact cu o frecvență stabilită de rata de transfer la recepție/transmisie pentru interfață serială dublă.

Sistemul de întreruperi este organizat pe 8 nivele prioritare și este realizat cu 8259A. Trei nivele de întrerupere sînt utilizate în cadrul modului de bază, celelalte fiind disponibile pentru modulele de extensie.

Nivelul 0, prioritate maximă, este alocat canalului 0 al contorului programabil care furnizează o întrerupere periodică utilizată pentru ceasul de timp real al sistemului. Nivelul 1 este alocat tastaturii. Nivelul 6 este alocat discului flexibil.

Cele 8 nivele sînt puse în corespondență cu nivelele de întrerupere 8—15 ale microprocesorului.

Configurația de bază a sistemului include 4 canale de acces direct la memorie. Canalul DMA 0 este utilizat pentru operațiile de reîmprospătare a memoriei RAM dinamice de pe placa de bază. Cererile de reîmprospătare sînt lansate la fiecare 15  $\mu$ s de către contorul 1. În acest fel numai 7% din timpul sistemului este consumat cu operația de reîmprospătare. Canalul DMA 2 este utilizat pentru operațiile cu discul flexibil. Canalele DMA 1 și 3 sînt disponibile la conectorii de extensie.



Modulul de bază este structurat în jurul a două magistrale:

- magistrala sistemului care permite cuplarea extensiilor;
- magistrala locală care cuplează resursele locale ale sistemului.

### Considerații asupra terminalelor color utilizate pentru FELIX-PC

Spre deosebire de dispozitivele de afișare pe tub catodic (CRT) monocrome (de obicei alb/negru), tuburile color utilizează un ecran acoperit cu mai multe tipuri de fosfor. Principiul de funcționare al tuburilor color se bazează pe faptul că diferite tipuri de fosfor emit lumină colorată diferit. Combinația dintre lumina emisă de mai multe tipuri de fosfor poate genera o întreagă paletă de culori diferite.

În construcția tuburilor pentru televizor se utilizează o mască perforată foarte fin pentru a obține culorile. Ecranul este acoperit cu triade foarte fine din fosfor de tipuri diferite. Astfel un punct din triadă emite culoarea roșie (R), al doilea emite culoarea verde (G), iar al treilea emite culoarea albastră (B). Cele trei puncte sînt dispuse foarte apropiate unele de altele, formînd o rețea de triade care generează punctele pe ecran (pixeli). Un tub color este echipat cu trei tunuri electronice, cîte unul pentru fiecare punct din triadă. Maska perforată este plasată în spatele ecranului acoperit cu triade de fosfor și are funcția de focalizare a celor trei tunuri electronice, astfel ca fiecare fascicul să fie focalizat numai pe punctul asociat din fiecare triadă. Controlînd intensitatea fiecărui fascicul se comandă intensitatea emisiei luminoase a fiecărui punct din triadă, obținîndu-se puncte de diferite culori pe ecran.

Dacă fiecare fascicul electronic este controlat binar (el poate fi activ sau inactiv, i.e. aprins sau stins) se obțin următoarele culori:

R	G	B	culoare	denumire în engleză
0	0	0	negru	black
0	0	1	albastru	blue
0	1	0	verde	green
0	1	1	turcoaz	cyan
1	0	0	roșu	red
1	0	1	purpuriu	magenta
1	1	0	cafeniu	brown
1	1	1	alb	white

După modul în care se transmit la tubul catodic informațiile de aprindere/stingere a punctelor pe ecran se disting următoarele tipuri de dispozitive de afișare pe tub catodic:

- monitor RGB;
- monitor cu intrare video-compus;
- televizor color.

### Monitor RGB

Informațiile privind activarea celor trei tunuri electronice pentru fiecare triadă se transmit direct de la interfața cu monitorul. Pentru a putea funcționa corect interfața trebuie să furnizeze monitorului semnalele de sincronizare. Sincronizarea se poate face prin semnale separate pe orizontală (HSYNC) și pe verticală (VSYNC) sau printr-un semnal de sincronizare compus (CSYNC =



HSYNC+VSYNC). Deoarece semnalele pentru controlul intensității tunurilor electronice se transmit direct, la astfel de monitoare imaginea este de cea mai bună calitate.

### Monitor cu video-compus

Aceste monitoare primesc informațiile de sincronizare, luminozitate și culoare codificate într-un semnal denumit video-compus. Acest semnal este decodificat de monitor pentru a obține, separat, informațiile necesare intern. Datorită codificării în interfață și decodificării în monitor calitatea imaginii va fi scăzută față de monitoare RGB. De asemenea, este necesar ca atât codificarea cât și decodificarea să respecte același standard (PAL, SECAM sau NTSC cu variantele acestora), altfel monitorul va afișa doar imagini alb-negru.

### Televizor color

Pentru afișarea color a imaginilor se poate utiliza un televizor color obișnuit dacă semnalul vide-compus generat de interfață este utilizat pentru a modula o purtătoare de radio frecvență, cu frecvența corespunzătoare unuia din canalele VHL sau UHF disponibile la televizor. Televizorul va recepționa prin antena semnalul de radiofrecvență, va face demodularea pentru a obține semnalul video-compus pe care îl va decodifica apoi pentru a obține informațiile de sincronizare, strălucire și culoare necesare pentru a comanda cele trei fascicule electronice ale tubului. Prin operațiile de codificare-modulare-demodulare-decodificare calitatea semnalului se deteriorează obținând astfel performanțe mai scăzute față de monitoare video-compus sau RGB. De asemenea banda de trecere a amplificatorului video de la televizor este o limitare în ceea ce privește frecvența punctelor trimise de interfață.

La microcalculatorul FELIX-PC se poate conecta un monitor cu intrare video-compus (standardul PAL sau NTSC) sau un televizor color, utilizând un modulator de radio frecvență extern pentru canalul dorit. De asemenea FELIX-PC poate utiliza un monitor RGB cu două trepte de intensitate (RGBI), obținându-se în acest caz încă 8 culori, deci 16 culori în total. În acest caz cele 16 culori posibile sint:

I	R	G	B	culoare	denumire în engleză
0	0	0	0	negru	black
0	0	0	1	albastru	blue
0	0	1	0	verde	green
0	0	1	1	turcoaz	cyan
0	1	0	0	roșu	red
0	1	0	1	purpuriu	magenta
0	1	1	0	cafeniu	brown
0	1	1	1	alb	white
1	0	0	0	gri	gray
1	0	0	1	bleu	light blue
1	0	1	0	verde deschis	light green
1	0	1	1	turcoaz deschis	light cyan
1	1	0	0	roșu aprins	light red
1	1	0	1	purpuriu deschis	light magenta
1	1	1	0	galben	yellow
1	1	1	1	alb strălucitor	light white

Pentru a genera cele 16 culori monitorul RGB trebuie să permită intrări pentru R, G, B, I și SYNC.



## Rezoluția și culorile la FELIX-PC

Interfața pentru terminalul grafic color constituie un modul de extensie standard în configurația microsistemului FELIX-PC. Această soluție a fost aleasă pentru a asigura o mai mare flexibilitate în ceea ce privește subsistemul de afișare grafică. Adaptorul pentru monitor conține 32 Kocteți de memorie biport organizată ca „bit mapped display”. Memoria de reîmprospătare a ecranului este inclusă în spațiul de adresare al procesorului. Acest lucru permite o flexibilitate maximă în elaborarea programelor de manipulare a imaginilor pe ecran.

FELIX-PC poate să funcționeze cu monitoare RGBI și sincronizări (VSYNC HSYNC) separate, cu monitoare color cu intrare video-compus sau televizoare color în sistemele NTSC (3, 54 Mhz) sau PAL.

Pentru a fi posibilă utilizarea, atât a unor monitoare performante, cât și a televizoarelor obișnuite este necesar ca rezoluția cu care se afișează imaginea să aibă cel puțin două trepte. Modurile de lucru, rezoluția și culorile sînt prezentate în următorul tabel:

mod	rezoluție	format grafic	format text	culori
1	mică	—	40×25	8 — fond 16 — caracter 16 — chenar 8 — fond
2	mare	—	80×25	16 — caracter 16 — chenar 16 — fond
3	mică	320×200	40×25	3 — pt. punct (din 6 video) 3 — pt. punct (din 7 RGB)
4	medie	640×200	80×25	1 — pt. punct (din 16)

Modurile 1 și 2 permit lucrul în regim alfanumeric, care de fapt este un regim semigrafic avînd în vedere ca pe lîngă setul de caractere ASCII standard mai există o serie de caractere grafice care permit realizarea unor grafice de rezoluție mică adecvate pentru multe aplicații. În aceste moduri memoria de reîmprospătare a ecranului conține codurile ASCII ale caracterelor iar generarea caracterelor se face printr-un generator hardware. Atributele de culoare sînt asociate caracterelor și prin alegerea adecvată a culorilor de fond și a culorilor caracterelor se pot obține grafice sau texte multicolore.

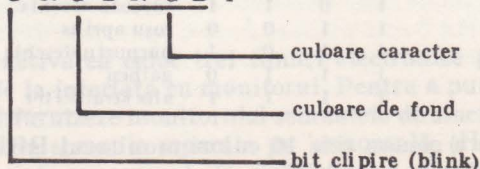
În modul alfanumeric (semigrafic) fiecare caracter este reprezentat în memoria de reîmprospătare pe 2 octeți. Primul octet conține codul ASCII al caracterului iar al doilea octet conține atributele de culoare și efecte speciale.

F E D C B A 9 8 7 6 5 4 3 2 1 0

cod ASCII

atribute

C R G B R G B I





Culoarea pentru chenar se selectează separat, într-un registru separat destinat acestei funcții.

Toți biții din octetul de atribute comandă culori separate care apoi sînt combinate de circuitele video din interfață pentru a forma culorile de fond și de desenare.

Modurile 3—6 permit lucrul în mod grafic, dar desigur permit și afișarea de texte. Memoria de reimprospătare corespunzătoare a ecranului conține informația care se afișează direct pe ecran. Deci în acest caz caracterele alfanumerice (și semigrafice) sînt generate prin program și ele apar ca grafice în memoria de ecran.

Correspondența dintre punctele de pe ecran și informația din memoria de ecran este:

— 1 bit / punct (pixel) pentru modul 4;

— 2 biți / punct (pixel) pentru modul 5.

Pentru modul 4 culorile de fond și chenar sînt întotdeauna negre iar culoarea cu care se desenează poate fi oricare din cele 16 culori posibile.

Pentru modul 3 există 3 palete a câte 3 culori pentru monitoarele RGB și 2 palete pentru monitoarele video și TV. Avînd în vedere că se poate selecta și una din cele 16 culori pentru fond, se pot obține desene multicolore formate practic din 4 culori simultan pe ecran. Codificarea culorilor pentru modul 3 este prezentată în continuare:

B1	B0	culoare (paleta 0)	culoare (paleta 1)	culoare (paleta 2)
0	0	culoare de fond	culoare de fond	culoare de fond
0	1	verde (green)	turcoaz (cyan)	turcoaz (cyan)
1	0	roșu (red)	purpuriu (magenta)	roșu (red)
1	1	cafeniu (brown)	alb (white)	alb (white)

Paleta 2 poate fi selectată numai pentru monitoarele RGB și nu este selectabilă prin BIOS. Pentru folosirea acesteia utilizatorul trebuie să-și prevadă rutinele corespunzătoare.

Pentru modurile alfanumerice (1 și 2), în memoria asociată terminalului grafic se pot organiza 8—respectiv 4 pagini ecran. Dintre acestea este activă (afișată pe ecran) una singură la un moment dat. Selectarea paginii de lucru se face prin programare.

### Sistemul de programe de bază și aplicații

Sistemul de programe implementat pe FELIX-PC are la baza sistemele de operare PC-DOS și MS-DOS și include:

— utilitarele sistemului de operare pentru interfața cu operatorul, gestiunea și întreținerea fișierelor, programe de test etc.

— facilități de execuție și depanare ale programelor;

— translatoare pentru programe în limbaj de asamblare și pentru limbajul BASIC;

— interpretor BASIC cu facilități pentru prelucrări grafice;

— mediu pentru dezvoltarea programelor în PASCAL; C; PROLOG

— mediu pentru dezvoltarea programelor în EDISON; MODULA-2

— programe de aplicații pentru:

— proiectare asistată de calculator;

— editarea și prelucrarea textelor;



- baze de date;
- culegerea și validarea datelor;
- prelucrări grafice;
- aplicații economice.

Datorită soluțiilor tehnologice ce vizează implementarea sistemului este de așteptat ca fiabilitatea sistemului să fie ridicată constituind o alternativă pentru diverse aplicații industriale.

Compatibilitatea cu microsisteme similare cu o largă răspindire cum ar fi: IBM-PC XT, IBM PS/2 MODEL 30 SANYO-550, OLIVETTI M24, CORONA, etc., oferă o mare disponibilitate software sistemului.

Tabelul 25.1

CARACTERISTICILE CALCULATOARELOR FELIX-PC și IBM PS/2-30 SÎNT:

	FELIX-PC	IBM PS/2-
Microprocesor	8086	8086
Frecvența de lucru	4,77 MHz	8 MHz
Coprocessor	8087 <sup>1)</sup>	8087
Număr de biți pe magistrala de date	16	16
RAM standard	256 Ko	640 Ko
Capacitate maximă RAM	640 Ko <sup>2)</sup>	640 Ko
ROM standard	12 Ko	64 Ko
Capacitate maximă ROM	96 Ko	64 Ko
Număr conectori	8 <sup>3)</sup>	3
Tip conector	IBM-PC	IBM-PC
Sistem operare	DOS 3.0 (DOS 3.3)	DOS 3.3
Dimensiunea și capacitatea discului flexibil	5" 1/4 <sup>4)</sup> 720 Ko	3" 1/2 720 Ko
Capacitatea discului Winchester	20 Mo <sup>5)</sup>	20 Mo
Moduri grafice standard	CGA, Hercules EGA <sup>6)</sup>	MCGA

<sup>1)</sup> opțional.

<sup>2)</sup> extensie pe placă separată

<sup>3)</sup> 4 conectori cu 2×31 contacte, 4 conectori cu 2×31 + 2×10 terminale.

<sup>4)</sup> unele serii sînt echipate cu unități de discuri flexibile simplă față, dublă densitate.

<sup>5)</sup> livrabil din 1988.

<sup>6)</sup> în curs de asimilare în 1988—89.

## 25.2. Noua familie de calculatoare personale IBM PS/2

### 1. Generalități.

Familia de calculatoare personale IBM PS/2 a fost anunțată în prima jumătate a anului 1987. Ea include cinci modele bazate pe microprocesoarele: 8086 (modelul 30-021), 80286 (modelele: 50-021 și 60-041) și 80386 (modelele: 80-041 și 80-111).



Toate modelele utilizează: unități de discuri flexibile de 3"1/2 (cu posibilități de conectare externă a unei unități de 5"1/4), controlor grafic pe placa de bază, circuite logice specializate (*custom logic chips*), tehnologia montajului pe suprafață (*surface mounted technology*) și o tehnică relativ simplă de asamblare. Construcția modulară asigură creșterea fiabilității de circa trei ori față de produsele anterioare și o reducere considerabilă a timpului de asamblare.

Sistemele PS/2 dispun de dispozitive video-color analogice și de o tastatură asemănătoare tastaturii perfecționate folosite la modelul anterior AT (cu excepția a trei diode luminescente pentru cele trei taste „Shift-Lock”).

Toate modelele au prevăzute pe placa de bază (mother board): circuitele video, portul serial, portul paralel, portul pentru dispozitivul indicator, ceasul care indică timpul. Ele se pot conecta la un monitor analogic RGB sau monocrom. De asemenea, toate calculatoarele se pot interfața cu o unitate de disc optic IBM, de 200 Mo (extern pentru modelele 30/50 și intern pentru modelele 60/80). Opțional este prevăzut un dispozitiv indicator de tip „mouse”, cu două butoane.

Firma nu oferă: listingul pentru PS/BIOS, schemele și caracteristicile electrice ale noilor rețele de porți. Vor fi puse la dispoziția utilizatorilor: punctele de intrare în BIOS și semnificațiile semnalelor electrice la noii conectori de extensie. Caracteristicile modelelor 30—80, din familia PS/2 sînt date în tabelul 25.2

Tabelul 25.2.

Caracteristicile modelelor din familia PS/2

Modelul	30-021 (8530-021)	50-021 (8550-021)	60-041 (8560-041)	80-041 (8580-041)	80-111 (8580-111)
Caracteristici					
Microprocesor	8086	80286	80286	80386	80386
Frecvența	8 MHz	10 MHz	10 MHz	16 MHz	20 MHz
Coprocesor	8087	80287	80287	80387	80387
Număr de biți de date/ magistrală	16	16	16	32 (11)	32 (11)
RAM standard	640 Ko	1 Mo	1 Mo	1 Mo	2 Mo
Cap. max. RAM	640 Ko (1)	7 Mo	15 Mo	16 Mo	16 Mo
ROM standard	64 Ko	128 Ko	128 Ko	128 Ko	128 Ko
Nr. conectori	3	4 (2)	8 (2)	8 (2)	8 (2)
Tip conector	IBM-PC	Micro-Channel	Idem	Idem	Idem
Sistem Operare	DOS 3.3	DOS 3.3; OS/2	Idem	Idem	Idem
Dimensiunea și Capaci- tatea discului flexibil	3" 1/2 720 Ko (3)	3" 1/2 1,44 Mo (4)	3" 1/2 1,44 Mo (5)	3" 1/2 1,44 Mo (5)	3" 1/2 1,44 Mo (5)
Capacitatea discului Winchester	20 Mo	20 Mo	44 Mo (6,7)	44 Mo (10)	115 Mo (12)
Capacitatea maximă a discului Winchester	(3)	20 Mo	88 Mo (7)	88 Mo (7)	230 Mo
Moduri grafice standard	MCGA	VGA, EGA, MCGA	Idem	Idem	Idem
Utilizarea displayului color IBM 8541, 1024×768	Nu	Da, cu placă opțio- nală ocupă un conector	Idem	Idem	Idem



Amplasarea unității de bază	pe masa „desktop“	Idem	În dulap vertical, „deskside“	Idem	Idem
Preț sistem	\$2295	\$3595	\$5295 (10)	\$6995 (10)	\$10995 (10)
Perfecționări majore față de sistemul din stînga	nu este cazul	80286, memorie mai mare, conector Micro-Channel, VGA	memorie mai mare, mai mulți conectori	80386 conectori de 32 biți	Ceas de 20 MHz, mai multă memorie RAM pe placa de bază

- (1) Se poate extinde memoria folosind plachete de memorie cu specificații Lotus/Microsoft Intel.
- (2) Interfața de disc Winchester ocupă un conector.
- (3) O versiune a modelului 30 (30-002) are o unitate de disc flexibil de 3" 1/2, în locul discului Winchester.
- (4) Modelul 50 dispune de spațiu pentru o altă unitate de disc flexibil de 1,44 Mo.
- (5) Modelul dispune de spațiu pentru o unitate de disc flexibil 3" 1/2, 1,44 Mo și pentru o unitate de disc Winchester sau optic suplimentare.
- (6) Modelul 60, sub forma 8560-071, este identic cu 8560-041, cu excepția unității de disc Winchester, de 70 Mo, și a memoriei RAM de 2 Mo.
- (7) Modelul 071 folosește pentru unitatea de disc Winchester interfața ESDI, la 10 MHz, care este de două ori mai rapidă decât interfața ST 506, folosită pentru unitatea de 44 Mo.
- (8) Modelul 60-071 poate avea maximum 185 Mo în unitatea de disc Winchester.
- (9) Prețurile nu includ sistemele de operare (\$ 125—\$ 795) și nici manualele (\$ 45—\$ 125).
- (10) Modelul 80, la 16 MHz, se livrează și sub forma 8580-71, identică cu 8580-041, cu excepția unității de disc Winchester, cu capacitatea de 70 Mo.
- (11) Toate modelele 80 au 8 conectori, dintre care 3 acceptă plachete de 16 sau 32 de biți; un conector de 16 biți este folosit de disc Winchester.
- (12) Atît unitatea de disc Winchester standard, cît și cea opțională, din modelul 80-111, folosesc interfața ESDI și sînt comandate la 10 Mbiți/s.

## 2. Hardware-ul de bază.

### 2.1 Modelul 30.

În modelul 30, microprocesorul 8086 operează la 8 MHz, fără stări de așteptare. El dispune de 640 Ko de memorie internă și de două unități de discuri flexibile de 3"1/2, cu capacitatea de 720 Ko fiecare.

În sistem sînt prevăzuți trei conectori de extensie tip IBM-PC, montați orizontal-lateral, pentru a ocupa un spațiu mai restrîns.

O serie de circuite, care la modelele anterioare se aflau pe plachetele de extensie, sînt plasate pe placa de bază: o versiune perfecționată a adaptorului grafic color IBM (CGA — *Color Graphic Adapter*), porturile serial/paralel, ceas/calendar și cuplorul de unitate de disc flexibil.

Datorită magistralei de date de 16 biți și a circuitelor suport, pentru 8086, modelul 30 poate opera la o viteză de 2,5 ori mai mare decît IBM PC-XT.

### 2.2 Modelele 50, 60 și 80.

Modelele 50 și 60, bazate pe microprocesorul 80286, cît și modelul 80, bazat pe microprocesorul 80386, au toate caracteristicile modelului 30, cu observațiile următoare: ele folosesc o magistrală nouă numită Micro-Channel, dispun pe placa de bază de un set de resurse hardware pentru un nou standard grafic, mai bun decît EGA (*Enhanced Graphic Adapter*), folosesc unități de discuri flexibile de 720 Ko.



Sistemele utilizează un cuplor de mare viteză, cu un factor de suprapunere 1:1, față de 3:1, în cazul calculatoarelor AT (Advanced Technology). Acest factor de suprapunere a fost posibil datorită modului de transfer în „rafală” oferit de noua magistrală. Modelele 60 și 80 dispun de o interfață perfecționată pentru dispozitivele mici (ESDI — Enhanced Small Device Interface). Cu aceste elemente, modelele 60 și 80 oferă o rată de transfer, pentru date, de șase ori mai mare decât IBM PC-AT.

Modelele 50 și 60 sînt aproape identice, cu unele deosebiri privind dimensiunea și capacitatea memoriei (1 Mo). Ambele folosesc microprocesorul 80286, care operează la 10 MHz, avînd ca opțiune coprocesorul matematic 80287, și sînt prevăzute cu noua magistrală Micro-Channel.

Modelul 50 este prevăzut cu o unitate de disc Winchester de 20 Mo, în timp ce modelul 60, în cele două versiuni: 60-041 și 60-071, are unități de discuri Winchester de 44 Mo și — respectiv 70 Mo.

Se apreciază că: folosind un ceas mai rapid, noua magistrală și un cuplor mai rapid de unitate de disc, aceste sisteme vor opera la o viteză dublă față de IBM PC-AT.

Modelul 80, care folosește versiunea standard Intel a microprocesorului B1 — 80386, este echipat cu 1—2 Mo de memorie (cu circuite de 1M bit), cu versiunea de 32 de biți a magistralei Micro-Channel. Este plasat într-un dulap vertical (*desktop*) și dispune de șapte conectori de extensie, dintre care trei sînt pentru plachete de 32 de biți.

Modelul 80, versiunea 041, este echipat cu o unitate de disc Winchester, de 44 Mo, în timp ce, pentru modelul 80, versiunea 071, discul are 70 Mo. Ambele versiuni au un ceas cu frecvența de 16 MHz. A treia versiune, 111, a modelului 80, dispune de un ceas de 20 MHz, 2 Mo de memorie internă și un disc Winchester cu capacitatea de 115 Mo. Modelul 80 este de 3,5 ori mai rapid decât IBM PC-AT.

### 3. Noile standarde video.

Familia IBM PS/2 încorporează trei noi standarde video. Modelul 30 utilizează sistemul MCGA (*Multi Color Graphics Adapter*), iar modelele 50, 60 și 80 folosesc sistemul VGA (*Video Graphic Adapter*) avînd ca opțiune monitorul de mare rezoluție 8514/A.

Sistemul MCGA constă din: două arii specializate de porți, 64 Ko de memorie RAM biport, generator de caractere bazat pe o memorie RAM statică de 16 Ko și un circuit specializat care implementează o paletă de culori de  $256 \times 18$  biți.

Sistemul suportă mai multe moduri:

- text: 80 coloane (cu rezoluție totală de  $640 \times 400$  pixeli, caractere de  $8 \times 16$  pixeli, 16 culori din 256 posibile),
- grafic:  $640 \times 200$  (cu două culori din 256 posibile și caractere de  $8 \times 8$  pixeli),
- grafic:  $640 \times 480$  (cu două culori din 256 posibile și caractere de  $9 \times 16$  pixeli).

În fiecare mod se poate folosi oricare din cele 262144 culori posibile. Deși rezoluția în modul grafic  $320 \times 200$  este redusă, numărul mare de culori (256) disponibile simultan pe ecran și numărul mare de culori posibile (262144) oferă sistemului posibilitatea unor efecte grafice deosebite.



Sistemul MCGA (*Multi Color Graphic Adapter*) poate emula vechiul sistem CGA, pentru programele care îl folosesc, dar nu poate emula EGA (*Enhanced Graphic Adapter*), fără o plachetă adaptoare specială.

Sistemul VGA este standard pentru modelele 50, 60, 80 și opțional, pentru modelul 30. VGA se bazează pe o arie de porți ce conține 12750 porți. În plus, față de modurile suportate de MCGA, VGA asigură atât standardul grafic EGA, cât și următoarele moduri suplimentare:

- text: rezoluție  $720 \times 400$  pixeli, caractere de  $9 \times 16$  pixeli;
- grafic: rezoluție  $640 \times 480$ , cu 16 culori din 256 posibile.

VGA arbitrează operațiile între memoria video și procesor, pe de-o parte și memoria video și circuitul de afișare, pe de altă parte.

Adaptorul grafic opțional 8514/A folosește conectorul video auxiliar al sistemului: un conector cu 20 de terminale în linie cu unul din conectorii de 16 biți ai magistralei Micro-Channel.

O plachetă video, plasată în acest conector, poate furniza pixelii săi sau/și semnalele de sincronizare în locul celor de pe placa de bază, oferind posibilitatea de a produce diferite moduri de afișare grafică, fără a înlocui circuitele video de pe placa de bază. Astfel, 8514/A asigură modul grafic de mare rezoluție:  $1024 \times 768$  pixeli, folosind noul monitor IBM — 8514. Cu ajutorul unei plachete opționale, de extensie a memoriei, se pot afișa 256 culori, din 262144 posibile, la rezoluția de mai sus. La o rezoluție mai mică:  $640 \times 480$  pixeli, adaptorul 8514/A oferă noi facilități constând în fonte programabile, pentru caractere, spațierea proporțională și umplerea cîmpurilor aparținînd unor forme.

#### 4. Monitoare analogice.

Sistemele grafice MCGA și VGA pot opera cu oricare din următoarele noi tipuri de monitoare analogice: monitor monocrom, de rezoluție medie, două monitoare color de rezoluție medie și un monitor color de mare rezoluție. Fiecare monitor are o frecvență de reimprospătare pe orizontală de 31,75 kHz și de 50 sau 70 Hz, pe verticală.

Modelul 8503 — monitor monocrom, are o diagonală de 12" și folosește versiunea IBM de luminofor alb (hîrtie), fără fenomene de licărire (*flicker*) sau instabilitate.

Adaptoarele MCGA și VGA pot detecta automat prezența acestui monitor, pentru a genera o imagine monocromă, folosind componenta verde, de 6 biți, a semnalului video, pentru a obține 64 nuanțe de gri.

Monitorul color 8512 are o diagonală de 14", în timp ce monitorul 8513 are o diagonală de 12" și pixeli de o finețe mai mare (0,28 mm).

Monitorul color 8514 are o diagonală de 16" și este compatibil cu sistemul VGA și MCGA, asigurînd o rezoluție de  $640 \times 480$  pixeli, precum și cu sistemul 8514/A, în cazul unei rezoluții de  $1024 \times 768$  pixeli.

#### 5. Compatibilitatea BIOS.

Noul BIOS (*Basic Input Output System*) este plasat în memorii ROM, în toate modelele seriei PS/2. În implementarea lui s-a urmărit compatibilizarea cu vechiul BIOS, la nivelul punctelor de intrare. Aceasta permite ca programele deja existente, care apelează funcțiile sistemului, chemînd rutine din BIOS, să se poată executa pe calculatoarele PS/2.



Modelul 50 este compatibil, la nivelul de BIOS, cu IBM PC-XT, PC Portabil, PC Convertibil și cu numeroase interfețe hardware existente. Scopul urmărit a fost acela de a păstra compatibilitatea cu produsele software existente pentru PC. Programele, care conțin rutine, în care intervine viteza de execuție a instrucțiunilor, nu vor opera corect, deoarece modelul 30 are un ceas de 8 MHz, față de 4,77 MHz, la vechile sisteme PC.

Modelele 50, 60, 80 dispun de un superset de PC-BIOS, numit CBIOS (*BIOS Compatibil*). Acesta poate adresa pînă la 1 Mo de memorie și permite execuția software-lui existent, cu foarte puține excepții.

BIOS-ul perfecționat ABIOS (*Advanced BIOS*) asigură suportul pentru operarea multitasking, adresînd pînă la 16 Mo de memorie.

BIOS-ul pentru modelul 30 este plasat în două circuite 27256, în timp ce, la modelele 50 și 60 este conținut în patru circuite 27256, ocupînd 128 Ko. Fiecare modul ROM are un octet de identificare, plasat la adresele F000 — FFFE.

În raport cu BIOS-ul cunoscut s-au modificat unele elemente privind întreruperile. Astfel, întreruperile OB(H) și OC(H), pentru comunicații, OD(H) și OF(H), pentru imprimantă, sînt rezervate. Întreruperea 15(H), care a fost asociată cu extensiile de sistem (*System Extensions*), pentru I/E caseta, deși îndeplinește aceeași funcție a fost denumită „Systems Services“. Întreruperile 40(H), 41(H), 46(H) și 4A(H), care au fost rezervate, controlează acum vectorul BIOS pentru dischetă, parametrii pentru discul fix și alarma utilizator. Întreruperile 71(H) — 74(H), 76(H) și 77(H) sînt rezervate. Ele au fost notate anterior, în zecimal, cu IRQ9 — IRQ12, IRQ14 — IRQ15. Întreruperile F1(H) — FF(H), care anterior nu au fost folosite, sînt acum rezervate ca Întreruperi-Program-Utilizator.

## 6. Noua magistrală „Micro-Channel“.

Noua magistrală nu păstrează compatibilitatea cu magistrala IBM-PC. Ea asigură: o viteză mai mare de transfer pentru date și I/E, partajarea resurselor și suportul necesar pentru multiprelucrare.

Modelele 50 și 60 folosesc varianta de magistrală de 16 biți, în timp ce modelul 80 utilizează magistrala de date de 32 de biți, pe 3 din cei 8 conectori.

Micro-Channel nu este multiplexat. Datele și adresele folosesc linii separate. Există linii suplimentare pentru comanda transferului, arbitrare, semnale suport și alimentare. Toate semnalele logice sînt compatibile TTL. Conectori magistralei au 2×58 terminale, organizate în două secțiuni: 2×46 (de la 01 — 46) și 2×12 (de la 47 — 58) și permit cuplarea plachetelor de 11" 1/2×3" ale interfețelor terminalelor (Tab. 25.3).

Tabelul 25.3.

Vedere din spate a plăcii de bază a sistemului

	B	A	
AUDIO GND .....	01 .....		—CD SETUP
AUDIO .....	02 .....		MADE 24
GND .....	03 .....	GND	
14,3 MHz OSC .....	04 .....		A11
GND .....	05 .....		A10
A23 .....	06 .....		A09







## MADE 24

M/-IO

—SO, —S1

—CMD

—CDSFPBK (n)

—CD.CHRDY (n)

CHRDYRTN

ARB0 ... ARB3

—PREEMPT

—BURST

—TC

—IRQ3 ... —IRQ7

—CD SETUP (n)

—CHCK

AUDIO

AUDIO GND

OSC

CRESET

—REFRESH

(1) Indică linii dedicate, separate pentru fiecare conector.

Bit ce indică prezența unei adrese de celulă de memorie, plasată în primii 16 Mo ai spațiului de adresare.

Bit ce specifică prezența unui ciclu de memorie sau de I/E.

Biți de stare ce definesc tipul ciclului de magistrală: scriere/citire.

Bit de comandă, care specifică data validă pe magistrală.

Bit de reacție la selecția plăchetei (1).

Bit indicator de canal pregătit (1).

Bit de retur de la canal pregătit.

Biți arbitrare/-acordare, folosiți pentru arbitrarea accesului la magistrală.

Semnul indicator de cerere de acces la magistrală.

Semnul indicator de folosire a magistralei în modul rafala.

Semnul asociat numărului terminalului.

Semnale indicatoare de cereri de întrerupere de la modulele „slave” de I/E.

Semn de activare a plăchetei (1);

Semn indicator de eroare în sistem.

Semn audio 50 Hz ... 10 KHz.

Masa semnalului AUDIO.

Semn de ceas, 14, 31818 MHz.

Semn RESET Canal.

Semn de reîmprospătare.

Atât magistrala, cât și întreaga arhitectură a sistemului au fost proiectate în vederea reducerii interferențelor electromagnetice. Astfel, la fiecare grup de patru terminale, ale conectorului, pe fiecare latură, un terminal este conectat la masă. Placa de bază și plăchetele interfețelor au planuri de masă și de alimentare.

Toate sistemele din familia PS/2 se plasează în clasa B-FCC, din punctul de vedere al radiațiilor electromagnetice, cu toate că au un ceas de frecvență ridicată.

Micro-Channel folosește protocoale asincrone pentru controlul magistralei și al transferurilor de date. Liniile cu semnalele -BURST și -TC controlează transferul datelor sub forma de blocuri. Un transfer de date se poate realiza fie cu memoria, fie cu un dispozitiv de I/E, în funcție de starea liniei asociate semnalelor M/-IO. Operațiile de citire și scriere, din cadrul transferurilor sînt definite de stările liniilor -SO și -S1. Semnalele de la liniile: PREEMPT, ARB/-GT și ARB0 pînă la ARB3 asigură arbitrarea magistralei.

Pentru a preîntîmpina pierderea unor cereri de întrerupere, în cazul proceselor concurente, s-au folosit întreruperi sensibile la nivel și nu la fronturi. Cererile de întrerupere sînt memorate în circuite bistabile, pînă la luarea lor în considerație. În acest mod crește și imunitatea la zgomet.

Magistrala folosește 11 linii de întreruperi prioritare. Ele sînt organizate, de la prioritatea cea mai mare, pînă la prioritatea cea mai mică, după cum urmează: -IRQ9 pînă la IRQ12, -IRQ14 pînă la -IRQ15 și -IRQ3 pînă la IRQ7.

Fiecare poziție (conector) posedă: cîteva linii care nu sînt partajate (ele sînt linii dedicate, care merg la fiecare poziție), precum și alte cîteva linii care sînt folosite pentru a indica lungimea informației în cadrul transferului de date (16 sau 32 biți; transferul pe 8 biți nu necesită un semnal special), linia de răspuns la selecția plăchetei (indică prezența plăchetei în conectorul dat, la o anumită adresă) și liniile de activare ale plăchetelor, folosite pentru selecția plăchetelor, în cadrul operației de configurare a sistemului.



Magistrala dispune de facilități care permit reducerea sau mărirea vitezei de operare a sistemului. Folosind aceste posibilități se pot proiecta plachete care pot opera pe magistrala Micro-Channel, fie în sistemele cu 80286, la 10 MHz, fie în sistemele cu 80386, la 16 sau 20 MHz.

Liniile AUDIO și GND AUDIO (masa audio) asigură posibilitatea conectării la ieșirea audio a sistemului sau pot fi folosite ca o cale de semnal audio între plachetele perifericelor. Frecvența de lucru a liniei este cuprinsă între 50 Hz și 10 KHz.

Accesul la magistrală este coordonat de un dispozitiv de arbitrare, localizat pe placa de bază. O cerere de acces la magistrală, din partea unei plachete, este efectuată prin activarea liniei -PREEMPT. În momentul în care placheta ce deține controlul magistralei termină operația, schema de arbitrare lansează un ciclu de arbitrare (se comandă linia ARB/-GNT), iar plachetele ce solicită controlul magistralei activează liniile de arbitrare ARBO până la ARB3. Nivelul afectat unei plachete date, pentru arbitrare, este fixat la pornirea sistemului, de către programul de configurare a sistemului. Placheta cu nivelul cel mai mic de arbitrare (cu cea mai mare prioritate) obține accesul la magistrală, când schema de arbitrare activează linia ARB/-GNT. Logica de arbitrare acordă fiecărei plachete intrate în competiție șansa de a avea acces la magistrală, înainte de a se intra într-un nou ciclu de arbitrare. Mecanismul de arbitrare este identic cu cel folosit pentru arbitrarea canalelor DMA.

Pentru configurarea sistemului, la pornire, se folosește un program special POS (Programmable Option Select), în cadrul căruia fiecare plachetă are codul ei de identificare. Din cele 64000 coduri posibile, IBM și-a rezervat jumătate, restul vor fi puse la dispoziția celor care vor fi implicați în dezvoltarea de aplicații. La pornirea sistemului (boot-up), se determină ce plachete sînt prezente, rezultatul fiind comparat cu lista plachetelor stocată într-o memorie RAM nevolatilă, la sfîrșitul sesiunii anterioare de lucru. Dacă sistemul nu detectează plachete noi, se vor încărca registrele de selecție, de pe plachete, cu datele din memoria RAM nevolatilă. Dacă sistemul descoperă o nouă plachetă (sau nu reușește să recunoască vreo plachetă), el va dezactiva placheta și va asigura execuția unei utilitare de configurare, care va permite asignarea resurselor sistemului noii plachete. În acest mod placheta asociată cu un periferic ce nu funcționează corect nu va rămîne cuplată on-line, evitîndu-se deteriorarea sistemului.

## 7. Noile sisteme de operare.

Familia de calculatoare PS/2 poate folosi sistemele de operare PC-DOS 3.3 (disponibil în 1987) și OS/2 (livrabil din 1988).

DOS 3.3 reprezintă o versiune perfecționată a sistemului DOS 3.2, în sensul că rezolvă o serie de probleme, apărute în versiunea precedentă. Astfel, poate adresa un disc Winchester cu o capacitate mai mare de 32 Mo, divizîndu-se în mai multe partiții. Versiunile anterioare permiteau mai multe partiții, dar PC-DOS putea adresa numai una din ele.

Au fost introduse noi comenzi pentru a satisface necesitățile unor medii de dezvoltare sofisticate.

Comanda APPEND permite unui program să găsească mai ușor fișiere în alte „subdirectories“. CALL oferă posibilitatea unui fișier „batch“ să execute un



## BIBLIOGRAFIE GENERALĂ CONSULTATĂ ȘI RECOMANDATĂ

1. Adrian Petrescu, Francisc Iacob, **Microcalculatorul personal HC-80** (Familia aMIC, Felix Student și HC-80), AMC vol 49, pag. 249, Editura Tehnică 1985.
2. \* \* \* **HC-85 Manual de utilizare**, Întreprinderea de Calculatoare Electronice, București, 1985.
3. Mike James, **An Expert Guide to the Spectrum**, Granada, London, Toronto, Sidney, New York, 1982.
4. S. M. Gee, **The Spectrum Programmer**, Granada, London, Toronto, Sidney, New York, 1982.
5. Mike James, **The Art of Programming the Spectrum**, Bernard Bahani — Publishing, 1983.
6. Dr. Ian Logan, Dr. Frank O'Hara, **Complete Spectrum ROM Desassembly** Melbourne House, 1983.
7. Ian Sinclair, **Introducing Spectrum Machine Code**, Granada, London, Toronto, Sidney, New York, 1983.
8. Adrian Petrescu, Gh. Rizescu și colectiv, **Totul despre aMIC**, Editura Tehnică, București, 1985.
9. Lance Leventhal, **Z 80 Assembly Language Programming** Osborne/McGraw-Hill, Berkeley, USA, 1979.
10. Elizabeth A. Nichols, Joseph C. Nichols, Peter R. Roni, **Z80 Microprocessor, Programming & Interfacing**, Book 1, Book 2, Howard W., Sams & Co., Inc. USA, 1979.
11. Adrian Dickens, **Spectrum Hardware Manual**, Melbourne House Publisher, UK, 1983.
12. \* \* \* **MEGABASIC** Sportscene Specialist Press Ltd, UK, 1984.
13. S. J. Wainwright, A. Grant, **Basic & Fortran per Spectrum**, Milano, Italia, 1984.
14. Dr. Tim Longdell, **43 ZX Spectrum Compiler. Owner's manual**, Softek Software, UK, 1984.

alt fișier „batch“, pentru ca apoi să-și continue execuția. Comanda FASTOPEN reprezintă o rutină care crează un tampon de nume-fișiere, permițând accelerarea redeschiderii fișierelor recent folosite. În acest mod cresc performanțele calculatorului, când operează cu fișiere, care sînt deschise și închise frecvent (fișierele manipulate într-o rețea).

OS/2 este un sistem multitasking, care adresează pînă la 16 Mo de memorie. El va încorpora unele caracteristici ale pachetului „Systems Application Architecture“, care va reprezenta o interfață comună, la nivelul tuturor sistemelor IBM. Ulterior el va avea o interfață bazată pe ferestre, grafică și simbolii. În final, ediția extinsă (*Extended Edition*) va include o bază de date relațională și funcțiuni de comunicație.

Prima ediție a OS/2 (Ediția Standard 1.0) a apărut în trimestrul întii al anului 1988, fără grafică și ferestre.

Următoarea ediție (Ediția Standards 1.1) va conține modulul „Manager Presentation“, care va include: grafica, ferestre, caractere de calitate tipografică și simbolii. „Presentation Manager“ este bazat pe pachetul „Microsoft Window“ și pe interfața „Graphical Data Display Manager“ (GDDM), de pe sistemele IBM 3270.

O altă versiune OS/2 (Ediția Extinsă) va avea toate facilitățile Ediției Standard, la care se vor adăuga: o versiune a pachetului de manipulare a bazelor de date relaționale IBM-DB2 și o versiune a limbajului structurat de interogare IBM-SQL. În plus, va conține o ediție evoluată a managerului de comunicații, în cadrul căruia utilizatorii vor comunica concurent prin intermediul mai multor protocoale.



15. David Link, **HISOFT PASCAL 4 V.1.4**, 13 Gooseacre, Cheddington, Beds, UK, 1984.
16. Liviu Dumitrascu, **Microelectronica interactivă**, Totul despre BASIC Editura Tehnică, București, 1989.
17. Pascal Pellier, **Lanfrage Machine Trues et Astuces sur ZX Spectrum**, Eyrolles, Paris, France, 1984.
18. Intr. Electronica (colectiv) CIP, **Calculator de instruire personal**, AMC. vol. 56—57, Editura Tehnica (sub tipar)
19. David Webbs, **Advanced Spectrum Machine Language**, Oxford, UK, 1984.
20. \* \* \* **ZX Interface and ZX Microdrive** Sinclair Research Limited., UK 1983.
21. \* \* \* **FORTH Editor's manual**, Sinclair ZX Spectrum, Computing, UK, 1983.
22. Dr. Ian Logan, **Il libro del microdrive Spectrum** Edizioni ICE, Milano, 1984.
23. Mike Lord, **Exploring Spectrum Basic**, Timedata Ltd, UK, 1982.
24. Dilwyn Jones, **Beyond Simple Basic Delving Deeper into your ZX Spectrum**, Interfaces Publications, London, — UK, 1983.
25. Jean-Francois Sehan; **Clefs pour le ZX Spectrum et Timex 2000**, Edition du PSI, Paris, 1983.
26. Tim Hortnell, **Le grand livre du ZX Spectrum**. Eyrolles, Paris, 1984.
27. David Link, **GENS 3 MONS 3. HISOFT 3T ZX Spectrum**, DEVPAC, 1983.
28. \* \* \* **Tasword Two The Word Processor**, Tasman Software, Leeds, UK, 1983.
29. Steven Vickers, **ZX SPECTRUM-Sinclair. BASIC Programming**. Sinclair Research Ltd, London, UK, 1983.
30. E. Sparer, **Sinclair Logo 1. Turtle Graphics** Sinclair Research Ltd, Cambridge, UK, 1984.
31. E. Sparer, **Sinclair Logo 2. Programing Reference Manual**, Sinclair Research Ltd, Cambridge, UK, 1984.
32. H. Abelson, **Logo for the Apple II**, Byte Books, McGraw-Hill, 1983.
33. H. Abelson, A. diSessa, **Turtle Geometry: The Computer as a Medium for Exploring Mathematics**, MIT Press, 1981.
34. K. Bowles, **Problem Solving Using Pascal**, Springer-Verlag, New York, 1977.
35. A. diSessa, **Unlearning Aristotelian Phisics: A Study of Knowledge-Based Learning**, Cognitive Science, 1981
36. A. J. Goldberg, D. Robson, D.M.H. Ingallas, **Smalltalk-80: The Language and its Implementation**, Addison Wesley, Reading, Ma, 1981.
37. P. Goldenberg, **Special Technology for Special Children**, University Park Press, Baltimore 1979.
38. J.A.M. Howe, T.O. Shea, F. Lone, **Teaching Mathematics through Logo Programming An Evaluation Study**, Department of Artificial Intelligence, University of Edinburg 1977.
39. A. Kay, **Microelectronics and the Personal Computers**, Scientific American, Sept. 1977.
40. S. Papert, **Mindstorms: Children Computers and Powerfull Ideas**, Basic Books, New York, 1980.
41. S. Papert, A. diSessa, D. Watt, S. Weir, **Final Report to the Brookline Logo Project: Assessment and Documentation of a Children's Computer Laboratory**, MIT Logo Project, Cambridge, MA, 1980.
42. D. Thornburg, **Discovering Apple Logo, An Invitation to the Art and Pattern of Nature** Addison-Wesley, 1983.
43. D. Watt, **Learning With Apple Logo**, Byte Books, McGraw-Hill, 1983.
44. P. Winston, B. Horn, **Lisp**. Addison-Wesley, MA, 1981.
45. \* \* \* **Apple Logo II: An Introduction to Programing** Logo Computer Systems Inc, 1984.
46. \* \* \* **Apple Logo II: Reference Manual**, Logo Computer Systems, Inc, 1982.
47. Gh. Sabău, N. Sotir și colectiv **Practica bazelor de date**, Editura Tehnică, 1989.
48. \* \* \* **Seria continuă AMC**, Editura Tehnică 1984—1989
49. A. Petrescu **Microprocesorul 8086**, AMC evol. 56—57, Editura Tehnică 1990 (sub tipar)
50. Bărbat I, Bărbat B, **Logica matematică și algebra booleană**, manual ptr. clasa a IX-a, E.D.P. București, 1978.
51. Bărbat I., Dumitrache A., **Matematica aplicată în tehnica de calcul**, manual ptr. clasa a IX-a, E.D.P., București, 1982.
52. Becheanu M., Căzănescu V., Năstăsescu C., Rudeanu S., **Logica matematică și teoria mulțimilor**, E.D.P., București, 1972.
53. Drăgănel T. **Complemente de matematică**, E.D.P., București, 1970.
54. Enescu Ghe., **Logica simbolică**, Editura Științifică, București, 1971.
55. Florica T. Cimpan, **Probleme celebre din istoria matematicii**, Ed. Albatros 1976.
56. Georg Klaus, **Logica modernă**, Ed. Științifică și Enciclopedică, București, 1977.



57. Moisil C. Gr., **Elemente de logică matematică și teoria mulțimilor**, Ed. Științifică, București, 1966.
58. Moisil C. Gr., **Scheme cu comandă directă cu contacte și relec**, Ed. Tehnică, București, 1964.
59. M. Lovin, P. Preoteasa, C. Popovici, **Informatica**, manual pentru anul II liceu — clase speciale de informatică, E.D.P., București, 1972.
60. Petrescu Adrian, **Calculatoare automate și programare**, E.D.P. București, 1973.
61. Quine W. V., **The Problem of Simplifying of Truth Functions** din Amer. Math. Monthly, 59, 8, 1952.
62. Reghiș M, **Elemente de teoria mulțimilor și logica matematică**, Editura Facla, Timișoara, 1981.
63. R. L. Morris și J. R. Miller, **Proiectarea cu circuite integrate T.T.L.**, traducere, Editura Tehnică, București, 1974.
64. Rîzescu Gh., Rîzescu Eugenia, **Teme ptr. cercurile de matematică din licee**, E.D.P. București, vol. I—1977, vol. II—1980.
65. Stănescu Ilie, Rădoi Ion, **Complemente de matematică ptr. cercurile de elevi din licee**, E.D.P., București, 1972.
66. Săndulescu I., Beloiu D., **Introducere în informatică**, manual ptr. clasa a IX-a, E.D.P., București, 1978.
67. Coman Gh., Frențiu M., **Introducere în informatică**, Ed. Facla, Timișoara 1982.
68. Stelian Niculescu, Mihai Radu Dumitrescu, **Algoritmi și metode de reprezentare** E.D.P., București, 1978.
69. Stelian Niculescu, **Ghid pentru predarea noțiunilor de informatică în învățământul pre-universitar**, București, 1987, M.E.I. I.C.S.I.T.—T.C.I.
70. Stelian Niculescu, Gh. N. Rîzescu, **Aspecte metodice privind predarea învățarea lecțiilor din capitolul Algoritmi**, articole în „Învățământul liceal și tehnic profesional”, Nr. 11/1987 și Nr. 12/1987.
71. Tomescu I., Leu A., **Matematica aplicată în tehnica de calcul**, manual — clasa a IX-a, E.D.P., București, 1980.
72. Virgil Brișcă, Bucur Ionescu, Ghe. Tudor, **Calcul numeric**, E.D.P., București, 1972.
73. Zvi Konavi, **Switching and Finite Automate**, ed. II, 1978, 1970, Mc. Graw Hill, Book Co, Computer Science Series.
74. \* \* \* **Programa de matematică ptr. învățământul de zi și seară**, aprobată cu Nr. 39732/1987, E.D.P., București, 1987.
75. Diac Florin, Gh. N. Rîzescu, Radu Jugureanu, Virgil Ionescu, Virgil Necula, Robert Al. Eckstein, Emil Mavrodin, Stelian Niculescu, ing. Andrei Polihroniade, Radu Bălan, **Buletin de informare metodico-științifică ptr. predarea informaticii în licee** vol. I.1988 și vol. II.1989, I.S.H.B;
76. Toacșe, Ghe. **Introducere în microprocesoare**, Ed. Științifică și Enciclopedică, București, 1985.
77. Gh. N. Rîzescu, **Teme ptr. folosirea noțiunilor de informatică în licee** lucrare realizată ptr. școlile Ministerului Industriei Ușoare — anul școlar 1987—1988.
78. N. Patrubby **Totul despre ... microprocesorul Z 80**, Editura Tehnică, 1989
79. Ahl. H. David, **BASIC Computer Games**, Morristown, New Jersey, 1981.
80. Benard J, **50 programmes ZX spectrum**, Editions Radio, Paris, 1983.
81. Brown R. Jerald **Instant (Freeze-Dried Computer Programming in BASIC)** London, 1982.
82. Dumitrașcu L., **GBASIC pentru începători cu calculatorul personal AMC nr. 48**, Editura Tehnică, București, 1986.
83. McIntire Thomas **The A to Z book of Computer Games**, USA, 1979.
84. Tachell J. et Bennett B, **Guide Hachette du microordinateur**, Hachette Paris, 1983.
85. **BASIC Programing of the Amstrad**. Micro Press, 1985.
86. \* \* \* **Dicționar de informatică**, Editura Științifică și Enciclopedică, București, 1981
87. \* \* \* **HC-85, Manual tehnic**, ICE, București, 1985.
88. \* \* \* **TIM S, Manual de funcționare și utilizare**, ITCI Timișoara, FMEDTC, 1987.
89. \* \* \* **ZX Spectrum Sinclair, Introduction**, Drim computers, 1986.
90. \* \* \* **Second International Conference: Children in the information age**, Sofia, Bulgaria, 1987 (Preprints).
91. Tibor Vasko, Darina Dicheva, **Educational policies: An international overview**, I. IASA Laxenburg, Austria, 1986.



92. UNESCO, *Education and Informatics*, Internațional congress, 12—21 April, 1989, Paris.
93. UNESCO, *USEIT Project*, Paris, 1989
94. UNESCO, *Informatics in Education*, Forth Conference of Ministers of Education, Paris 12—27 sept. 1988
95. Nicolae Teodorescu (Academician, Președintele Societății de Științe Matematice) „GA-ZETA MATEMATICĂ” anii 1974—1989.
96. Oleg Cernian, Paul Zamfirescu, *Limbajul BASIC pe calculatoarele WANG*, Editura Tehnică, 1984.

## ANEXE

### 1. METODICA REPREZENTĂRII ALGORITMILOR

Practica pedagogică de peste două decenii în procesul de predare-învățare, din cadrul laboratorului de tehnică de calcul și informatică al Liceului industrial „Dimitrie Cantemir”, din București, a evidențiat câteva aspecte cu privire la prioritatea în studiu a reprezentării structurale a algoritmilor.

În primul rând, pentru începători (și îndeosebi pentru vârsta școlară din gimnaziu și clasa a IX-a) însușirea reprezentării algoritmilor este optimizată prin procedeele schemei logice (scheme-bloc sau blocuri logice). Considerăm că motivația psiho-pedagogică a acestei constatări poate fi pusă în legătură cu accesibilitatea înținerii rapide și trainice a semnificației figurilor geometrice folosite în reprezentarea algoritmică a unei probleme.

Pe al doilea loc se poate situa reprezentarea algoritmilor prin **diagrame** (sub forma **arborescentă**). Această reprezentare, folosind o parte din formele geometrice de la blocurile logice, devine, de asemenea, accesibilă tineretului școlar în vîrstă de 10—15 ani. De aici, pe baza cunoașterii acestor reprezentări, se poate trece la scrierea într-un limbaj de programare mai simplu a programelor de rezolvare a unor probleme, spre exemplu, în **pseudocod**. Pe baza experienței, astfel acumulate, poate fi abordată scrierea „în direct” a unui program într-unul din limbajele de programare: BASIC, LOGO, PASCAL.

**Exemplu.** Pentru facilitarea comparării procedeele semnalate mai sus se va alcătui mai întîi schema logică și apoi diagrama corespunzătoare efectuării **împărțirii întregi a două numere naturale A, B, pe baza scăderii succesive**. În figurile 1 și 2, se evidențiază determinarea citului și a restului împărțirii lui A la B, folosind notații: „D” pentru diferență și „Q” pentru cit.

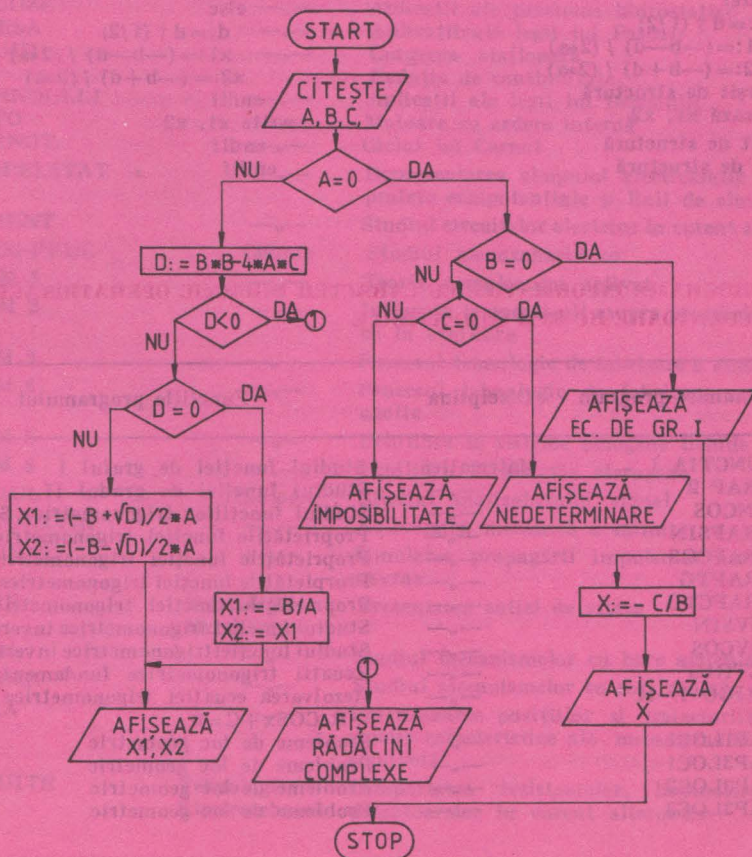
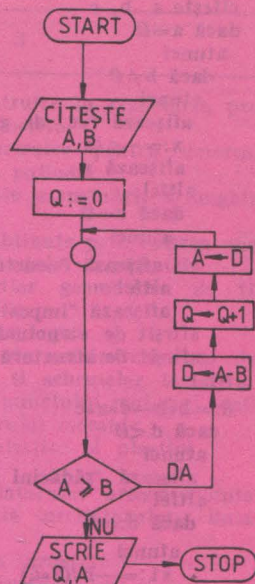
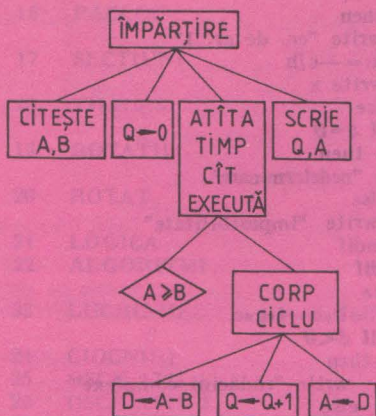
Vom observa faptul că într-o diagramă (fig. 2) operațiile se parcurg de sus în jos, pe nivele succesive, iar în cadrul aceluiași nivel — de la stînga la dreapta.

Rezultă că în acest proces algoritmic fiecare nou descăzut este rezultatul (restul) scăderii precedente, operația de scădere repetîndu-se pînă ce descăzutul devine mai mic decît scăzătorul. Descăzutul se micșorează deci treptat, în timp ce scăzătorul rămîne mereu același. Rezultatul ultimei scăderi efectuate este restul împărțirii celor două numere date, iar numărul de scăderi efectuate reprezintă citul împărțirii lui A la B.

**Alt exemplu.** În figura 3 este prezentată **schema logică** pentru rezolvarea în R a ecuației de gradul doi, avîndu-se în vedere atît cazul ecuației degenerate ( $a=0, b \neq 0$ ), cit și cazul de nedeterminare ( $a=0, b=0, c=0$ ).

În continuare este prezentată **scrierea în pseudocod** a programului de rezolvare a ecuației de gradul doi, în concordanță cu schema logică anterior realizată, instrucțiunile programului fiind prezentate și în limba engleză — prin scrierea în paralel a programului.







```

citește a, b, c
dacă a=0
atunci
    dacă b≠0
    atunci
        afișează "ec. de gr. I"
        x:=-c/b
        afișează x
    altfel
        dacă c=0
        atunci
            afișează "nedeterminare"
        altfel
            afișează "imposibilitate"
        sfârșit de structură
    sfârșit de structură
altfel
d:=b*b-4*a*c
dacă d<0
atunci
    afișează "rădăcini complexe"
altfel
    dacă d:=0
    atunci
        x1:=-b/(2*a)
        x2:=x1
    altfel
        d:=d↑(1/2)
        x1:=(-b-d)/(2*a)
        x2:=(-b+d)/(2*a)
    sfârșit de structură
    afișează x1, x2
sfârșit de structură
sfârșit de structură

```

```

read a, b, c
if a=0
then
    if b≠0
    then
        write "ec. de gr. I"
        x:=-c/b
        write x
    else
        if c=0
        then
            write "nedeterminare"
        else
            write "imposibilitate"
        endif
    endif
else
    d:=b*b-4*a*c
    if d<0
    then
        write "rădăcini complexe"
    else
        if d:=0
        then
            if d=0
            x1:=-b/(2*a)
            x2:=x1
        else
            d:=d↑(1/2)
            x1:=(-b-d)/(2*a)
            x2:=(-b+d)/(2*a)
        endif
        write x1, x2
    endif
endif
endif

```

## 2. PROGRAME INFORMATICE CU CARACTER DIDACTIC OPERATIONALE PE CALCULATOARE HC-85 ȘI COMPATIBILE

Nr. crt.	Denumire program	Disciplina	Funcțiile programului
1	FUNCTIA I	Matematică	Studiul funcției de graf 1
2	GRAF 2	—	Studiul funcției de gradul II
3	SINCOS	—	Studiul funcțiilor trigonometrice SIN și COS
4	GRAFSIN	—	Proprietățile funcției trigonometrice SIN
5	GRAFCOS	—	Proprietățile funcției trigonometrice COS
6	GRAFTG	—	Proprietățile funcției trigonometrice TG
7	GRAFTG	—	Proprietățile funcției trigonometrice CTG
8	INVSIN	—	Studiul funcției trigonometrice inverse ARCSIN
9	INVCOS	—	Studiul funcției trigonometrice inverse ARCCOS
10	ECTRIG	—	Ecuatii trigonometrice fundamentale
11	TRIGO I—II	—	Rezolvarea ecuației trigonometrice $A \cdot \sin x + B \cdot \cos x + C = 0$
12	CAP1LOC1	—	Probleme de loc geometric
13	CAP3LOC1	—	Probleme de loc geometric
14	CAP3LOC2	—	Probleme de loc geometric
15	CAP3LOC3	—	Probleme de loc geometric



0	1	2	3
16	PACLOC	—"	Program de autoinstruire în rezolvarea problemelor de loc geometric
17	SECTIUNI	—"	Program de autoinstruire pentru determinarea secțiunilor în poliedre
18	THALES	—"	Aplicații practice ale asemănării triunghiurilor
19	ROTATII	—"	Corpuri de rotație obținute prin rotirea unor figuri plane în jurul axelor de simetrie
20	ROTAT	—"	Studiul transformărilor geometrice de tip rotație
21	LOGICA	—"	Prezentarea operațiilor logice
22	ALGORITMI	—"	Program informatic de autoinstruire în domeniul algoritmilor și schemelor logice
23	LUCRU-MEC	Fizică	Energia mecanică a punctului material, definiția și calculul lucrului mecanic
24	CIOCNIRI	—"	Studiul ciocnirilor elastice și plastice
25	OSCILATII 1	—"	Miscarea oscilatorie
26	OSCILATII 2	—"	Legea mișcării oscilatorului armonic liniar
27	OSCILATII 3	—"	Viteza și accelerația oscilatorului liniar armonic
28	OSCILATII 4	—"	Energia oscilatorului armonic
29	OSCILATII 5	—"	Compunerea oscilațiilor
30	PENDUL	—"	Studiul pendulului gravitațional
31	ECLUZE	—"	Aplicații ale presiunii hidrostatice
32	PRESA	—"	Aplicații ale legii lui Pascal
33	FLUID	—"	Curgerea staționară
34	BERNOULLI	—"	Ecuția de continuitate
35	OTTO	—"	Aplicații ale legii lui Bernoulli
36	CARNOT	—"	Motoare cu ardere internă
37	CIMPELSTAT	—"	Ciclul lui Carnot
38	CURENT	—"	Reprezentarea cîmpului electrostatic prin suprafețe echipotențiale și linii de cîmp
39	GLUC-FRUC	Chimie	Studiul monozaharidelor
40	CHIM 1	—"	Teoria complexului activat
41	CHIM 2	—"	Influența temperaturii asupra proceselor chimice în echilibru
42	CHIM 3	—"	Procesul tehnologic de fabricare a amoniacului
43	CHIM 4	—"	Procesul tehnologic de fabricare a acidului azotic
44	CHIM 5	—"	Echilibre în sisteme omogene lichide
45	CHIM 6	—"	Echilibru în sisteme eterogene
46	OCHIUL	Biologie	Studiul analizatorului vizual
47	CIRCULATIA	—"	Activitatea mecanică a inimii
48	NEURON	—"	Simularea propagării impulsului nervos prin nevrax
49	CUTIE	Tehnologie-Mecanică	Prezentarea cutiei de viteze
50	TERN	—"	Studiul mecanismelor cu bare articulate
51	MECDIF	—"	Studiul mecanismelor cu roți dințate
52	BÎLEA	—"	Determinarea pozițiilor și traiectoriilor unor puncte caracteristice ale mecanismului bielă-manivelă
53	CIRCUITE	Tehnologie-Electrotehnică	Comportarea rezistoarelor, bobinelor, condensatoarelor în curent alternativ



0	1	2	3	0
54	GR. MOBIL	Tehnologie-Mecanică	Grade de mobilitate și scheme structurale la mecanismele plane	
55	GRAM 3	Limbi străine	Test de gramatică în limba germană	
56	VOC 3	— " —	Test de vocabular în limba germană	
57	TEXT 3	— " —	Test de înțelegere a unui text în limba germană	
58	VOCAB-FR	— " —	Test de vocabular în limba franceză	
59	GRAM-FR	— " —	Test de gramatică în limba franceză	
60	GRAM 9	— " —	Test de gramatică în limba engleză	
61	VOC 9	— " —	Test de vocabular în limba engleză	
62	TEXT 9	— " —	Test de înțelegere a unui text în limba engleză	

\* Programele, împreună cu documentația, se pot solicita la ICI București Secția de cercetare Sisteme și programe informatice pentru învățământ și instruire, B-dul Miciurin Nr. 8—10, sector 1, Tel. 656060/258, Telex 12891).

De asemenea, programe pentru folosirea calculatorului în procesul instructiv-educativ pot fi solicitate Ministerului Învățământului — Oficiul Central pentru Mijloace de Învățământ.

Programe pentru învățământ, precum și articole metodico-stiințifice de specialitate, se pot găsi în Gazeta Matematică și în publicațiile de specialitate ale Societății de Științe Matematice din România, în publicațiile Oficiului de Informare Documentară ale M.I., ale Centrelor de Calcul ale universităților și institutelor politehnice, ale inspectoratelor școlare și ale unor licee din țară.

Activități de instruire pentru învățământ desfășoară și multe Centre teritoriale de calcul, ca și o serie de Centre de calcul departamentale și uzinale.

Nu putem lăsa la o parte importanta activitate editorială din domeniu (în deosebi a Redacției de informatică și tehnică de calcul a Editurii Tehnice), care asigură existența în tiraje mari a unor cărți (inclusiv cu medii magnetice) în care se găsesc adevărate biblioteci de programe pentru învățământ (cum este cazul cu prezenta lucrare).



### 3: INTERPRETORUL BETA BASIC 3.1\*)

(instrucțiuni și funcții în plus față de BASIC HC-85)

#### 1. Generalități

Beta Basic 3.1 este un interpretor de BASIC care cuprinde un set de instrucțiuni și funcții mult extins față de interpretorul BASIC HC-85 (și compatibile), care este rezident în memoria EPROM.

Interpretorul Beta Basic 3.1 se încarcă de pe casetă în memoria RAM a calculatorului.

Prin setul său extins de instrucțiuni și funcții permite elaborarea sistematică a programelor.

Cuvintele cheie "rezervate" din Beta Basic 3.1 se pot obține în două moduri:

— în modul grafic "G" se apasă o tastă corespunzătoare fiecărui cuvânt cheie. Corespondența pe tastatură este:

DEFPROC	PROC	ENDPROC	RENUM	WINDOW	AUTO	DELETE	KEYWORDS	EDIT		
1	2	3	4	5	6	7	8	9	10	
DEFKEY	DEFAULT	LOCAL	KEYIN	EDIT	JOIN	REF	CSIZE			
POP		ELSE	ROLL	TRASE	FN	USING	EXIT	IF	ON	DPOKS
Q	W	E	R	T	Y	U	I	O		P
ALTER	SCROLL	DO	FILL	GET	BLANK	WHILE	UNTIL	LOOP		
A	S	D	F	G	H	J	K	L		CR
			CLOCK		BLANK	ON ERROR	SORT			
CS	Z	X	C	V	B	N	M	SS		BREAK

— în modul stabilit de KEYWORDS 3 sau KEYWORDS 4, în care instrucțiunile și comenzile se pot introduce tastând literă cu literă (ca la tastaturile calculatoarelor profesionale). Acest mod este recomandat, deoarece nu trebuie să se cunoască poziția cuvintelor cheie asociate instrucțiunilor și funcțiilor, pentru că introducerea lor se face literă cu literă.

Interpretorul Beta Basic folosește același spațiu de memorie pentru program și variabile ca și interpretorul BASIC HC-85 (cu excepția spațiului ocupat de interpretor în ultima zonă de memorie RAM). Dacă din anumite motive (aparitia unei erori netratate corespunzător sau efectuarea unor comenzi referitoare la discul flexibil care nu sînt implementate, cum ar fi CAT sau ERASE) se iese din interpretorul Beta Basic și se ajunge sub controlul Basic HC-85, se poate reveni în Beta Basic prin execuția comenzii:

RANDOMIZE USR 58419

fără a se pierde programul deja introdus. Dacă programul respectiv nu are instrucțiuni specifice numai lui Beta Basic atunci el poate fi executat și sub BASIC HC-85 (găsindu-se în spațiul de memorie gestionat de BASIC HC-85).

Limbajul BASIC implementat de interpretorul Beta Basic 3.1 oferă următoarele facilități principale:

- posibilitatea de a defini proceduri care au nume simbolic și parametri formali. Transferul de date între program și subrutină se realizează prin parametrii actuali, care se asociază parametrilor formali (ca în limbajul FORTRAN, PASCAL, etc.);
- apelarea procedurilor se realizează prin nume urmat de parametrii actuali;
- lucrează cu variabile globale (a căror valori sînt accesibile tuturor procedurilor) și cu variabile locale (a căror valori sînt accesibile numai procedurii care o folosește și numai pe durata cit procedurii este în execuție);
- permite editarea instrucțiunilor, cu posibilitatea de mutare a cursorului în toate direcțiile (stînga, jos, sus, dreapta);
- poate întrerupe și programe în cod mașină care ciclează;
- permite definirea de ferestre;
- permite afișarea ceasului cu posibilități de fixare a orei;
- permite definirea de caractere de diferite dimensiuni;
- asigură diverse moduri de lucru cu tastatura (introducerea cuvintelor cheie direct prin apăsarea unui grup de taste sau introducerea literă cu literă);



- conține instrucțiuni care asigură posibilitatea elaborării structurate a programelor:

DO-LOOP DO-UNTIL condiție DO-WHILE condiție EXIT-IF condiție  
și definire de proceduri cu nume simbolic cu DEFPROC - ENDPROC, etc.

- conține instrucțiuni de umplere a unui contur (FILL);
- are instrucțiuni care asigură decizie multiplă;
- permite poziționarea pe ecran la nivel de pixel a unui text (PLOT x, y; a\$);
- permite mutarea conținutului ecranului (sau o zonă de ecran) în toate direcțiile, cu un număr de pixeli specificat;
- permite renumerotarea instrucțiunilor din tot programul (sau a unor secțiuni de program);
- dispune de instrucțiuni care sortează șiruri sau elementele variabilelor indexate (crescător sau descrescător);
- permite apelarea recursivă a procedurilor;
- permite copierea (mutarea directă) a unui număr de elemente dintr-o variabilă în alta;
- permite listarea structurală (indentată) a întregului program (sau pe porțiuni) sau listare a unei proceduri;
- listează variabilele ce apar în program și permite editarea variabilelor numerice și de tip șir;
- listează numărul instrucțiunilor ce conțin o anumită referință (nume, număr, șir) și permite schimbarea unei referințe (nume de variabilă, atribut, etc.) cu o altă referință;

## 2. Sumarul comenzilor și instrucțiunilor Beta Basic 3.1 suplimentare față de BASIC HC-85

Obs: entitatea cuprinsă între simbolurile <...> reprezintă un parametru opțional.

**ALTER** <atribut> **TO** <atribut>

permite manipularea fișierului care conține atributele ecranului (INK, PAPER, FLASH, BRIGHT pentru fiecare caracter).

**ALTER** referința1 **TO** referința2

schimbă în tot programul referința1 cu referința2. Referința poate fi o variabilă, un număr sau un șir de caractere.

**AUTO** <număr linie inițială> <.pas>

numerează automat liniile "introduse de la tastatură" în timpul elaborării programelor.

**BREAK** (CAPS SHIFT și SPACE simultan)

Înterupe programul în curs de execuție (chiar și o rutină în cod mașină), cu mențiunea că programul în cod mașină să nu conțină dezactivare întreruperi.

**CAT** <număr. drive>

afișează numele fișierelor care se găsesc pe microdiver-ul sau discul flexibil specificat. În anumite implementări ale lui Beta Basic, această funcție nu funcționează pentru disc flexibil. Lansarea în execuție a unei comenzi CAT 1 va trece controlul lui BASIC HC 85. Aici se poate executa comanda CAT 1, după care se poate reveni în Beta Basic fără a se pierde programul. Revenirea se face cu RANDOMIZE USR 58419.

**CLEAR** număr

deplasează RAMTOP-ul cu un număr de locații specificat de număr.

"-767 < număr < 767"

**CLOCK** mod de lucru

stabilește modul de lucru al ceasului, care se poate afișa în dreapta sus a ecranului.

mod de lucru	salt la subrutină	alarmă audio	afișează timpul ora:minute:secunde
0	nu	nu	nu
1	nu	nu	da
2	nu	da	nu
3	nu	da	da
4	da	nu	nu
5	da	nu	da
6	da	da	nu
7	da	da	da



**CLOCK „hh:mm:ss“ „ahh:mm:ss“**

șirul de caractere specifică valoarea de început a ceasului (ora:minut:secunda) și poziționează timpul când alarma va deveni activă.

**CLS <număr\_fereastră>**

șterge fereastra cu numărul specificat sau fereastra curentă.

**COPY variabilă\_numerică1 <(slicer)> TO variabilă\_numerică2 <(poziție)>**

**COPY variabilă\_șir1 <(slicer)> TO variabilă\_șir2 <(poziție)>**  
copiază șiruri sau tablouri de date în întregime (sau elemente specificate de slicer) de la variabilă1 la variabilă2. Variabila sursă nu este afectată. Prin slicer înțelegem o specificație de forma <număr> TO <număr>.

**CSIZE lățime\_caracter <,înălțime\_caracter>**

modifică mărimea, în pixeli, a caracterelor. Poate avea un efect global dacă e folosită ca instrucțiune sau efect local dacă e folosită în PRINT sau PLOT.

**DEFAULT variabilă=expresie <,variabilă=expresie>...**

atribuie variabilei, valoarea rezultată în urma evaluării expresiei, numai dacă în timpul execuției instrucțiunii DEFAULT, variabila nu era deja creată.

**DEFAULT=dispozitiv**

stabilește dispozitivul cu care vor lucra în mod implicit comenzile LOAD și SAVE

**DEFKEY literă; șir**

stabilește tasta specificată de literă, atribuindu-i textul specificat prin șir. Activarea se face prin apăsarea simultană a tastelor ss+space+literă

**DEFPROC nume\_procedură <parametru1><,parametru2>...<,REF parametruj>.**

definește o procedură cu numele simbolic nume\_procedură cu parametri locali specificați de parametri și cu parametri globali specificați de REF parametri

**DELETE <număr\_linie\_început> TO <număr\_linie\_sfîrșit>**

șterge toate liniile de program din blocul specificat. Dacă se omite primul parametru se începe cu prima linie după linia 0, iar dacă se omite al doilea parametru se șterge pînă la sfîrșitul programului

**DELETE nume\_șir <(slicer)>**

**DELETE nume\_tablou\_date <(slicer)>**

șterge șirul de date, tabloul de date sau părți ale acestora. Acestea dispar complet sau își modifică dimensiunea. Slicer reprezintă o parte din elementele tabloului, specificată sub forma <număr1> TO <număr2>

**DO**

**DO WHILE condiție**

**DO UNTIL condiție**

mai chează secvența de instrucțiuni ce se va executa în ciclu. Sfîrșitul secvenței care definește corpul ciclului este marcat de LOOP.

**DO DO DO WHILE condiție DO UNTIL condiție**

.....

.....

**LOOP LOOP UNTIL condiție LOOP LOOP**

Din prima formă de specificare se iese cu EXIT IF condiție sau cu POP. A doua formă, execută corpul ciclului dintre DO și LOOP numai dacă condiția este falsă (practic se execută pînă cînd condiția este îndeplinită).

A treia formă, execută corpul ciclului atît timp cît condiția este îndeplinită (pînă cînd condiția devine falsă).

**DPOKE adresă, număr**

depune în memorie la adresă și adresă+1 valoarea numărului specificat (valoare organizată pe 2 octeți).

**DRAW TO x, y <, unghi>**

desenează segmente și arce ca și DRAW, numai că x și y reprezintă coordonate absolute.

**EDIT număr\_linie**

poziționează cursorul pe linia specificată și o aduce în zona de editare (partea de jos a ecranului) pentru a fi editată.

**EDIT variabilă\_șir**

**EDIT; variabilă\_numerică**

editează valoarea variabilelor specificate.

### 3. INTERPRETORUL BETA BASIC



## ELSE instrucțiune

face parte din instrucțiunea:

**IF condiție THEN instrucțiune ELSE instrucțiune**

Dacă condiția nu este îndeplinită se execută instrucțiunea specifică după ELSE.

## ENDPROC

marchează sfârșitul corpului de definire a unei proceduri. În urma execuției lui ENDPROC controlul programului se întoarce la instrucțiunea următoare apelului, eliberând spațiul ocupat de variabilele locale.

## ERASE <dispozitiv>; nume

șterge fișierul cu numele specificat de pe dispozitivul respectiv. În anumite implementări ale lui Beta Basic, aceasta funcție nu funcționează pentru disc flexibil. Lansarea în execuție a unei comenzi ERASE "d"; 1; "nume" va trece controlul lui BASIC HC-85. Aici se poate executa comanda ERASE "d"; 1; "nume", după care se poate reveni în Beta Basic fără a se pierde programul. Revenirea se face cu RANDOMIZE USR 58419

## EXIT IF condiție

este utilizată în cadrul unui ciclu DO...LOOP în vederea asigurării ieșirii din ciclu în momentul când condiția este îndeplinită. În momentul ieșirii din ciclu se continuă cu instrucțiunea de după LOOP.

## FILL <INK culoare>; x, y

## FILL <PAPER culoare>; x, y

umple o zonă de INK sau PAPER, delimitată de un contur, cu o culoare specificată, începând din punctul de coordonate x, y.

## GET variabilă numerică

## GET variabilă șir

reprezintă, un mod de a citi de la tastatură, asemănător cu INKEY\$, diferența constând în faptul că GET așteaptă apăsarea unei taste înainte de a continua.

## GET variabilă șir,x,y <, lățime, înălțime> <; tip>

introduce informația dintr-un dreptunghi de imagine într-o variabilă de tip șir. Această informație poate fi readusă pe ecran prin instrucțiunile PLOT sau CSIZE n <, m>. Informația din fiecare pătrat de 8x8 pixeli este reprezentată în șir prin 8 caractere (octeți) în formatul UDG

- x, y reprezintă coordonatele colțului din stînga sus (în pixeli);
  - lățime, înălțime, sînt dimensiunile dreptunghiului în caractere (8x8pixeli);
  - tip reprezintă tipul de preluare a informației
- 0 — fără atribute  
1 — cu atribute

## JOIN <număr linie>

concatenează linia cu numărul specificat cu linia următoare.

Aceasta din urmă va pierde numărul și va fi separată de linia precedentă prin „:“

## JOIN variabilă numerică1 <(slicer)> TO variabilă numerică2 <(poziție)>

## JOIN variabilă șir1 <(slicer)> TO variabilă șir2 <(poziție)>

are ca efect mutarea elementelor din variabila șir1 (sau din tabloul definit de variabila numerică1) în variabila2. Mutarea se poate face și parțial prin specificarea unor elemente din acestea, precizate de slicer, începând cu poziția specificată. Dacă lipsește slicer se consideră toate elementele, iar dacă lipsește poziția se ia lungimea destinației +1 (adăugare la sfîrșit); slicer reprezintă o specificare de forma <număr 1> TO <număr2>

## KEYIN variabilă șir

poate fi utilizată numai în program (nu și în mod comandă). Introduce variabila șir în program ca și cum ar fi introdusă de la tastatură, permițînd programelor să se poată automodifica

## KEYWORDS număr

număr=0

dezactivează posibilitatea de introducere de la tastatură a instrucțiunilor specifice Beta Basic, permițînd lucrul cu caractere semigrafice

număr=1

activează posibilitatea de introducere de la tastatură a instrucțiunilor specifice Beta Basic, (ce se introduc trecînd în mod grafic, urmat de o tastă specifică instrucțiunii — din § .1)



**număr=2**

cuvintele cheie ale instrucțiunilor Beta Basic se introduc doar în forma token (mod grafic+tastă specifică)

**număr=3**

cuvintele cheie ale instrucțiunilor Beta Basic se introduc fie sub formă „token” fie prin tastarea literă cu literă (alegerea celei de a doua posibilități se face prin apăsarea unui spațiu la începutul introducerii instrucțiunii curente)

**număr=4**

cuvintele cheie ale instrucțiunilor Beta Basic se introduc numai prin tastarea literă cu literă.

**LET variabilă1=expresie1, variabilă2=expresie2, ...**

constituie o dezvoltare a instrucțiunii LET din BASIC-HC85 prin faptul că permite mai multe atribuirii pe aceeași linie.

**LIST < linie\_inițială > TO < linie\_finală >**

listează secțiunea de program cuprinsă între cele 2 numere de linii specificate.

**LIST DATA**

listează toate variabilele.

**LIST VAL**

listează toate variabilele numerice — dînd valorile pentru cele reale și dimensiunile pentru cele indexate.

**LIST VAL**

listează toate variabilele de tip șir.

**LIST DEFKEY**

listează definițiile de taste.

**LIST FORMAT mod-listare**

mod listare — 0

listează ca în BASIC-HC85

mod listare — 1

listează textul structurat, indentînd corpul structurilor cu un spațiu spre dreapta;

mod listare — 2

listează textul structurat, indentînd corpul structurilor cu două spații spre dreapta;

mod listare — 3

listează ca în BASIC-HC85, fără etichetele liniilor;

mod listare — 4

listează textul structurat, indentînd corpul structurilor cu un spațiu spre dreapta, fără etichetele liniilor;

mod listare — 5

listează textul structurilor, indentînd corpul structurilor cu două spații spre dreapta, fără etichetele liniilor;

**LIST PROC nume\_procedură**

listează instrucțiunile care constituie corpul procedurii specificate (între DEF PROC și END PROC)

**LIST REF referință**

listează etichetele liniilor în care apare „referința”.

Referința poate fi:

— nume variabilă

— număr

— șir

**LOCAL variabilă1<,variabilă2>.....**

declară variabilele locale ale unei proceduri. Variabilele locale sînt asociate apelului procedurii respective și există numai pe durata execuției acesteia. Nu are nici o legătură cu eventualele variabile care au același nume în programul apelat sau în altă procedură. Parametrii formali dacă nu sînt declarați cu REF sînt considerați implicit variabile locale.

**LOOP**

**LOOP UNTIL condiție**

**LOOP WHILE condiție**

este o componentă a structurii DO-LOOP ce marchează sfîrșitul secvenței de instrucțiuni care se execută în ciclu. Instrucțiunea LOOP realizează saltul (direct sau condiționat) la instrucțiunea DO corespunzătoare.

**ON**

se utilizează în structuri ca:

**GOTO ON număr; număr linie1, număr linie2,...**

**GOSUB ON număr; număr linie1, număr linie2,...**

realizează saltul la instrucțiunea (sau subrutina) cu număr linie corespunzător, în funcție de valoarea numărului care poate fi și variabilă sau expresie. utilizată în forma:



**ON număr :instrucțiune1 : ..... : RETURN**

constituie o alternativă pentru GOSUB ON când subrutinele au o singură instrucțiune. În funcție de numărul specificat după ON (care poate fi și variabilă sau expresie) se execută una din instrucțiunile care urmează până la RETURN. utilizată în forma:

**ON ERROR nr.linie****ON ERROR ; instrucțiune1 : instrucțiune2 : ..... : RETURN**

realizează un GOSUB la număr linie specificat (sau la instrucțiunea cospunzătoare) în momentul apariției unei erori. Utilizându-se variabilele de sistem LINE, STAT, ERROR se poate localiza eroarea și tipul ei, în timpul execuției programului.

**OVER 2**

realizează SAU logic între pixelii existenți pe ecran și cei nou afișați (nu șterge și nu inversează nimic). În general este utilizat ca atribut temporal.

**PLOT x, y <; variabilă.șir>**

are o acțiune asemănătoare cu instrucțiunea PLOT x,y din BASIC HC-85, utilizată cu atributele uzuale INVERSE, OVER, INK, etc. la nivel de pixel.

În plus permite ca variabila de tip șir să se scrie pe ecran, fiind poziționată nu la nivel de linie și coloana ci la nivel de pixel.

**POKE adresă, variabilă.șir**

depune în memorie, începând cu adresa specificată valoarea variabilei șir (care poate fi și constantă). Este o extensie a instrucțiunii POKE adresă, număr.

**POP <variabilă.numerică>**

extrage din stivă conținutul vârfului stivei, ce poate constitui ultima informație depusă de GOSUB, DO-LOOP, PROC și îl atribuie variabilei specificată (dacă aceasta există). Este utilizată pentru a afla de unde a fost apelată o procedură. Poate servi și pentru a ieși dintr-un ciclu DO-LOOP.

**<PROC> nume procedură <parametru1><,parametru2>...<REF parametruj>...**

apelează procedura cu numele nume procedură furnizându-i parametrii actuali. Cuvântul cheie PROC poate lipsi, apelul realizându-se numai prin numele procedurii înscris în lista de parametri actuali.

În momentul apelării procedurii, interpretorul va crea cîte o variabilă de tipul specificat în lista de parametri formali, care va exista doar pe timpul execuției procedurii și îi va da valoarea pe care o are parametrul actual.

Pentru parametrul global (precedat de REF) nu se creează o nouă variabilă ci, un pointer (o adresă) către acea variabilă din programul apelant care constituie parametrul actual.

Parametrii de tip tablou trebuie să fie specificați cu REF parametru(), aceștia fiind întotdeauna variabile globale.

Se pot defini proceduri care să aibă un număr variabil de argumente, în acest caz lista de argumente înlocuindu-se cu DATA.

În Beta Basic se permite apelarea recursivă a procedurilor.

**READ LINE variabilă.șir1 <,variabilă.șir2>...**

citește date alfanumerice care apar în instrucțiunea DATA fără ca acestea să fie intercalate între ghilimele.

**REF referință**

caută în program referința specificată (care poate fi nume de variabilă, număr sau șir) și aduce în zona de editare linia care conține acea referință, cursorul fiind plasat imediat după referință.

La apăsarea lui CR se introduce linia în program (corectată sau nu). La tastarea lui CR se aduce în zona de editare, următoarea linie care conține referința, etc.

Căutarea se încheie cu introducerea ultimei linii în care apare referința.

Forma REF nume variabilă poate apare în definirea unei proceduri cu DEF PROC, în apelul procedurii sau în listarea unor referințe.

**RENUM <\*><(<nr.linie1> TO <nr.linie2>)><LINE nr linie nouă><STEP pas>**

renumerează liniile de program cuprinse în secțiunea delimitată de nr. linie1 și nr.linie2. Linia de început va primi numărul nr linie nouă iar pasul de numerotare va fi stabilit de pas. În forma simplă RENUM, se renumerează tot programul începând cu eticheta 10 și pasul 10.



Dacă nr liniei lipsește se consideră începutul programului. Dacă nr liniei2 lipsește se consideră sfârșitul programului. Dacă LINE nr linie nouă lipsește se consideră ca linie de început a renumerotării nr liniei.

Dacă STEP pas lipsește se consideră implicit pasul 10.

Simbolul \* specifică copierea zonei de program delimitată de nr liniei1 și nr liniei2 în poziția nr linie nouă renumerotată cu pasul specificat, fără a o șterge din poziția inițială.

Atenție la programele care au GOTO expresie.

**ROLL** cod.direcție <,pixeli> <;x, y; lățime, înălțime>

mută ecranul sau o zonă de ecran (o fereastră) specificată prin coordonatele colțului stînga sus (x, y) și lățime respectiv înălțime, în direcția indicată de cod direcție, cu un număr de puncte specificat de pixeli.

Informația ce iese prin una din marginile ferestrei apare în marginea opusă. În funcție de cod direcție se pot muta datele, atributele sau ambele informații cuprinse în fereastra respectivă, conform tabelului următor:

cod direcție	direcție (sens) mișcare	informație mutată
1	stînga	atributele
2	jos	atributele
3	sus	atributele
4	dreapta	atributele
5	stînga	datele
6	jos	datele
7	sus	datele
8	dreapta	datele
9	stînga	atributele și datele
10	jos	atributele și datele
11	sus	atributele și datele
12	dreapta	atributele și datele

**SCROLL** cod - direcție <,pixeli> <;x, y; lățime, înălțime>

are același efect cu ROLL numai ca informația care iese prin una din marginile ferestrei se pierde și nu mai apare prin marginea opusă.

**SAVE** <nr - liniei1> TO <nr - liniei2>; <dispozitiv> nume

salvează o zonă de program delimitată de nr liniei1 și nr liniei2 pe dispozitivul specificat (casetă, microdrive, disc flexibil), sub numele specificat.

**SAVE DATA** <dispozitiv> nume

salvează zona de variabile pe dispozitivul specificat "casetă, microdrive, disc flexibil" sub numele specificat în comandă.

**SORT** <INVERSE> variabilă - șir <(slicer)>

**SORT** <INVERSE> variabilă - indexată <(slicer)>

ordonează crescător sau descrescător elementele dintr-o variabilă șir sau dintr-un tablou de numere sau șiruri de date. Poate fi sortată și numai o zonă din tabloul specificat, delimitată de slicer. (slicer reprezintă: <număr1> TO <număr2>).

**SPLIT**

dacă o linie care cuprinde mai multe instrucțiuni, pe care o edităm va conține <> ca prim caracter la început de instrucțiune, atunci partea de dinaintea lui <> va fi introdusă în program iar restul va rămîne în linia de editare. <> va fi înlocuit cu o copie a etichetei originale a liniei.

**TRACE** număr - linie

asigură o facilităate de depanare a programelor BASIC. Are ca efect lansarea în execuție a subrutinei care începe cu numărul de linie specificat.

Forma:

**TRACE** :instrucțiune1: instrucțiune2: ... :RETURN

oferă aceleași facilități ca și forma anterioară numai că subrutina conține numai o singură linie.

**UNTIL**

utilizat în una din formele :

**DO UNTIL** condiție

**LOOP UNTIL** condiție

are rolul de a permite execuție condiționată a secțiunii de program cuprinsă între DO-LOOP, de un număr de ori pînă cînd condiția devine adevărată.



**USING format \_afişare; variabilă \_numerică**

permite specificarea formatului de afişare a valorii variabilei numerice. Astfel dacă format afişare este (###.##) atunci valoarea variabilei se afişează în formatul cu trei cifre partea întreagă şi cu două cifre partea zecimală.

**VERIFY (<nr - linie1> TO <nr - linie2>) ;><dispozitiv> nume**

verifică dacă o parte din program, delimitată de nr liniei 1 şi nr linie2, este identică cu zona corespunzătoare din fişierul specificat de pe dispozitivul indicat în comandă.

**VERIFY DATA <dispozitiv> nume**

verifică variabilele din program cu variabilele din fişierul specificat.

**WHILE**

utilizat în formele:

**DO WHILE condiţie****LOOP WHILE condiţie**

are rolul de a permite executare condiţionată a secţiunii de program, delimitată de DO-LOOP, de un număr de ori până când condiţia devine falsă.

**WINDOW nr - fereastră <, x, y, lăţime, înălţime>**

defineşte o fereastră pe ecran cu numărul nr fereastră, în poziţia specificată prin coordonatele (x, y) ale colţului stnga sus, şi de dimensiuni stabilite de lăţime şi înălţime. Se pot defini maxim 128 de ferestre (nr fereastră poate lua valori între 0 şi 127). Se pot defini atribute permanente în fiecare fereastră iar scroll-ul se realizează independent în cadrul fiecărei ferestre. WINDOW 0 reprezintă întreg ecranul.

**XOS, YOS****XRG, YRG**

reprezintă variabile speciale care permit schimbarea originii axelor şi a scării de reprezentare în cadrul execuţiei instrucţiunilor PLOT, DRAW, CIRCLE, FILL.

XOS, YOS reprezintă originea axelor (implicit 0,0);

XRG, YRG reprezintă dimensiunea axelor (implicit 256, 176).

**3. Sumarul funcţiilor Beta Basic 3.1 suplimentare faţă de BASIC HC-85****AND**

realizează SI logic între biţii celor două numere specificate (corespunzători acestoraşi poziţii în reprezentarea binară a numerelor). Numerele sînt cuprinse între 0 şi 65535.

**BIN\$ (număr)**

realizează conversia din zecimal în binar a numărului specificat. Rezultatul este furnizat sub forma de şir de caractere. Dacă numărul este mai mic decît 256 atunci şirul generat are 8 caractere, iar dacă numărul este mai mare de 255 şi mai mic decît 65535 şirul generat are 16 caractere.

**CHAR\$ (număr)**

converteşte numere mai mici de 65536 într-un şir de două caractere ASCII.

**COSE (număr)**

calculează cosinusul numărului, cu o precizie de 4 cifre semnificative. Lucrînd cu o precizie mai redusă este mult mai rapid decît funcţia standard "de circa 10 ori".

**DEC (şir - hexa)**

converteşte un şir de două sau patru caractere hexazecimale în numărul zecimal corespunzător.

**DPEEK (adresa)**

furnizează valoarea ce se găseşte în memorie în doi octeţi succesivi. La adresa specificată se găseşte, octetul cel mai puţin semnificativ iar la adresa +1 se găseşte partea cea mai semnificativă.

**EOF (număr - canal)**

furnizează informaţii referitoare la poziţia în cadrul unui fişier de date, de pe microdrive sau disc flexibil, în timpul operaţiilor de citire date. Astfel rezultatul furnizat de EOF este:

1 - cînd s-a citit ultimul element din fişier;

0 - cînd mai sînt date în fişier.

Atenţie, cînd se utilizează cu discul flexibil.



**FILLED ( )**

generează numărul de pixeli poziționați de ultima comandă FILL executată.

**HEX\$ (număr)**

realizează conversia numărului specificat, într-un șir de două sau patru cifre hexa. Dacă numărul este mai mic decât 256 se generează un șir de două caractere, iar dacă numărul este cuprins între 256 și 65535 se generează un șir de patru caractere. Reprezintă inversul operației DEC. Numerele negative sînt considerate în complement față de 2.

**INARRAY (variabilă șir\_indexată (poziție start <,slicer>), subșir)**

caută într-un tablou de șiruri de date (sau într-o parte specificată de slicer) un subșir, începînd cu poziția indicată de poziție start. Generează un număr ce specifică poziția primului element din tablou care conține subșirul specificat, sau 0 dacă nu se găsește pînă la sfîrșit.

O parte din caracterele subșirului pot fi #, ceea ce semnifică faptul că în locul lor poate fi considerat orice caracter. INARRAY lucrează cu tablouri cu două dimensiuni.

**INSTRING (poziție start, șir, subșir)**

caută dacă subșirul specificat în funcție se găsește în șirul specificat cu parametru, începînd cu poziție start.

Generează un număr ce specifică poziția primului element din șir care conține subșirul specificat, sau 0 dacă nu se găsește pînă la sfîrșit.

Șirul poate fi oricît de lung, în schimb subșirul poate avea maxim 255 caractere. O parte din caracterele subșirului pot fi #, ceea ce semnifică faptul că în locul lor poate fi considerat orice caracter.

**ITEM ( )**

este utilizată în operațiile de citire a datelor cu READ, DATA sau în cadrul procedurilor cu număr variabil de parametri (DEF PROC nume procedura DATA) Generează:

- 0 — dacă toate elementele listei curente DATA au fost citite;
- 1 — dacă următorul element este valoare alfanumerică;
- 2 — dacă următorul element este valoare numerică.

**LENGTH (cod funcție, variabila indexată ( )**

generată adresa din memorie sau dimensiunile tabloului specificat prin variabila indexată, în funcție de cod funcție. Astfel dacă:

cod funcție = 0 generează adresa în memorie unde se găsește tabloul;

cod funcție = 1 generează lungimea primei dimensiuni a tabloului;

cod funcție = 2 generează lungimea celei de a doua dimensiune a tabloului.

**MEM ( )**

furnizează numărul de octeți de memorie disponibili.

**MEMORY\$ ( ) (slicer)**

furnizează conținutul zonei de memorie specificată de slicer, sub forma de șir de caractere.

**MOD (număr1, număr2)**

generează număr1 modulo număr2 (echivalent cu restul împărțirii întregi a numerelor număr1 la număr2).

**NUMBER (șir)**

convertește un șir de două caractere într-un număr natural corespunzător.

**OR (număr1, număr2)**

realizează SAU logic între biții celor două numere specificate (corespunzători aceluiași poziții în reprezentarea binară a numerelor) Numerele sînt cuprinse între 0 și 65535.

**RNDM (număr)**

furnizează un număr aleator cuprins între 0 și numărul specificat. Față de RND din BASIC HC-85 este mai rapid.

**SCRN\$ (linie, coloană)**

generează codul ASCII al caracterului care se găsește pe ecran în poziția specificată de linie și coloană. Față de funcția SCREEN\$ care o regăsim și în BASIC HC-85 are avantajul că recunoaște și caracterele grafice.



## SHIFT\$ (cod \_operație, șir \_caractere)

modifică șirul de caractere specificat în funcție de cod operație Astfel:

cod operație = 1 transformă toate literele din șir în litere majuscule;

cod operație = 2 transformă toate literele din șir în litere mici;

cod operație = 3 transformă toate literele mari în litere mici și cele mici în majuscule;

cod operație = 4 toate caracterele de control devin ( ) cu excepția lui carriage return CR (CHR\$ 13);

cod operație = 5 transformă caracterele CHR\$ 128-127 în CHR\$ 0-127

și apoi realizează funcția de la cod operație = 4

cod operație = 6 transformă caracterele CHR\$ 128-128 în CHR\$ 0-127

și apoi realizează funcția de la cod operație = 4 inclusiv pentru CHR\$ 13;

cod operație = 7 transformă cuvintele cheie din forma „token” în formă literă cu literă;

cod operație = 8 transformă cuvintele cheie în forma „token” când le recunoaște (indiferent că sînt scrise cu litere mari sau cu litere mici și imediat după un cuvînt cheie nu trebuie să urmeze o literă);

cod operație = 9 realizează aceeași funcție ca la cod operație = 8, numai că după cuvîntul cheie poate urma o literă;

cod operație = 10 realizează aceeași funcție ca la cod operație = 9, numai că trebuie scrise cu majuscule;

cod operație = 11 realizează aceeași funcție ca la cod operație = 8, numai că trebuie scrise cu majuscule.

## SINE (număr)

calculează sinusul numărului, cu o precizie de 4 cifre semnificative. Lucrînd cu o precizie mai redusă este mult mai rapid decît funcția standard (de circa 6 ori).

## STRING\$ (număr, șir)

tipărește șirul specificat de un număr de ori indicat de numărul specificat ca parametru

## TIME\$ ( )

furnizează timpul curent ca un șir de 8 caractere

## USING\$ (format, număr)

generează un șir de caractere echivalent numărului specificat în formatul indicat de format (vezi instrucțiunea USING).

\* Bibliografie.

\* \* \* Beta Soft, Oxford, England, 1987.

\* \* \* Inf. nr. 1/1988, Buletin al clubului programatorilor, Casa universitarilor, Timișoara, 1988.



#### 4. Paragraful 16.5. de la pag. 100 din vol. 2 (CONTINUARE)

*Soluția problemei de la clasa a VII-a 1987*

```

10 INPUT "Nr. perechi (x, y)="; n
20 DIM x(n)
   DIM y(n)
   DIM e(n)
30 REM citește perechi
40 PRINT "Introduceți valorile perechilor"
50 FOR i=1 TO n
60 PRINT "x(";i;")=";
70 INPUT "x=";x(i)
80 PRINT x(i);" ";
90 PRINT "y("; i;")=";
100 INPUT "y=";y(i)
110 PRINT y(i); " ";
120 PRINT
130 NEXT i
140 REM calculează x min și x max
150 FOR i=1 TO n:
   LET e(i)=x(i):
NEXT i
160 GO SUB 430
170 LET xmin = min:
   LET xmax=max
180 REM calculează ymin și ymax
190 FOR i=1 TO n:
   LET e(i)=y(i):
NEXT i
200 GO SUB 430
210 LET ymin=min:
   LET ymax=max
220 REM ordonare după valorile x
230 LET indo=0
240 FOR i=1 TO n-i
250 IF x(i) < =x(i+1) THEN GO TO 290
260 LET indo=1
270 LET tx=x(i):
   LET x(i)=x(i+1):
   LET x(i+1)=tx
280 LET ty=y(i):
   LET y(i)=y(i+1):
   LET y(i+1)=ty
290 NEXT i
300 IF indo=1 THEN GO TO 220
310 REM calculul factorului de normalizare xn respectiv yn
320 IF xmin=xmax THEN LET xn=0:
   GO TO 340
330 LET xn=200/(xmax-xmin)
340 IF ymin=ymax THEN LET yn=0:
   GOTO 360
350 LET yn=150/(ymax-ymin)
360 REM trasare grafică
370 CLS
380 PLOT 0, (y(1)-ymin)*yn
390 FOR i=1 TO n-1
400 DRAW (x(i+1)-x(i))*xn, (y(i+1)-y(i))*yn
410 NEXT i
420 STOP
430 REM rutina de calcul a valorii minime și maxime dintr-un șir de numere
440 REM intrări - șirul de numere

```



```

450 REM ieşiri — min= valoarea minimă
460 REM — max = valoarea maximă
470 LET min=e(1)
    LET max=e(1)
480 FOR i=2 TO n
490 IF min > e(i) THEN LET min=e(i)
500 IF max < e(i) THEN LET max=e(i)
501 NEXT i
520 RETURN

```

*Soluția problemei de la clasa a VIII-a 1987*

```

10 READ ne
20 DIM n$(ne, 9):
    DIM p$(ne, 7):
    DIM e$(1, 9):
    DIM m(ne):
    DIM r(ne):
    DIM f(ne):
    DIM g(ne)
30 REM citire date despre elevi
40 FOR i=1 TO ne:
    READ n$(i), p$(i), m(i), r(i), f(i)
50 NEXT i
60 REM media notelor
70 FOR i=1 TO ne:
    LET g(i)=INT (100*(m(i)+r(i)+f(i))/3)/100:
    NEXT i
80 REM dialog cu utilizatorul.
90 CLS
100 PRINT "Alegeți una din funcții"
110 PRINT "1 — ordine alfabetică"
120 PRINT "2 — ordine descrescătoare medii"
130 PRINT "3 — reprezentare grafică note"
140 INPUT "opțiune="; o
150 IF o=1 THEN GO TO 190
160 IF o=2 THEN GO TO 290
170 IF o=3 THEN GO TO 370
180 GO TO 140
190 REM rutina de ordonare alfabetică
200 LET indo=0
210 FOR i=1 TO ne-1
220 IF n$(i) < n$(i+1) THEN GO TO 240
230 GO SUB 610
240 NEXT i
250 IF indo=1 THEN GO TO 190
260 GO SUB 700
270 INPUT "apăsați orice tastă"; q$
280 GO TO 80
290 REM opțiune 2 ordonare după medie
300 LET indo=0
310 FOR i=1 TO ne-1
320 IF g(i) > g(i+1) THEN GO TO 340
330 GO SUB 610
340 NEXT i
350 IF indo=1 THEN GO TO 290
360 GO TO 260
370 REM opțiune 3 reprezentare note elev
380 INPUT "introduceți nume elev .:"; e$(1)
390 REM caută elev
400 LET pe = 0
410 FOR i=1 TO ne
    IF e$(1)=n$(i) THEN LET pe=i

```



```

420 NEXT i
430 IF pe=0 THEN CLS:
    PRINT "nu există elev cu acest nume":
    GO TO 270
440 CLS
450 INK 2:
    LET nota = m(pe):
    PLOT 120,0:
    GO SUB 570
460 INK 3:
    LET nota = r (pe):
    PLOT 152,0:
    GO SUB 570
470 INK 4:
    LET nota = f(pe):
    PLOT 184,0:
    GO SUB 570:
480 INK 5:
    LET nota = g(pe):
    PLOT 216,0:
    GO SUB 570
490 INK 0
500 PRINT "Nume Prenume Mat Rom Fiz Med"
510 PRINT n$ (pe); p$ (pe);
520 PRINT m(pe); " ";
530 PRINT r(pe); " ";
540 PRINT f(pe); " ";
550 PRINT g(pe); " ";
560 GO TO 270
570 REM rutina pentru trasare dreptunghi proporțional cu nota
580 REM intrări nota
590 DRAW 24,0:
    DRAW 0, nota*15:
    DRAW -24,0:
    DRAW 0, -nota*15
600 RETURN
610 REM rutina de schimbare poziție 2 elevi
620 LET indo=1
630 LET t$=n$(i):
    LET n$(i)=n$(i+1):
    LET n$(i+1)=t$
640 LET t$=p$(i):
    LET p$(i)=p$(i+1):
    LET p$(i+1)=t$
650 LET t=m(i):
    LET m(i)=m(i+1):
    LET m(i+1)=t
660 LET t=r(i):
    LET r(i)=r(i+1):
    LET r(i+1)=t
670 LET t=f(i):
    LET f(i)=f(i+1):
    LET f(i+1)=t
680 LET t=g(i):
    LET g(i)=g(i+1):
    LET g(i+1)=t
690 RETURN
700 REM rutina de tipărire
710 REM intrări — datele despre elevi
720 REM ieșiri — afișează aceste date
730 PRINT "Nume Prenume Mat Rom Fiz Med"
740 FOR i=1 TO ne

```



```

750 PRINT n$(i);
760 PRINT p$(i);
770 PRINT m(i);
780 PRINT r(i);
790 PRINT f(i);
800 PRINT g(i)
810 NEXT i
820 RETURN
830 DATA 3
840 DATA "popescu", "ion", 8, 4, 3
850 DATA "ionescu", "petre", 4, 7, 8
860 DATA "manea", "dan", 9, 8, 7

```

## 16.5.2. Concursul național de informatică 12-23 iulie 1988

### PROBLEME PROPUSE

#### Clasa a V-a

Un grup de elevi formează o coloană care are  $m$  rînduri,  $m=15$ , cu  $n$  elevi pe rînd  $n=6$ . De pe fiecare rînd este ales cel mai scund, elev, iar dintre cei aleși, cel mai înalt primește un steag. Al doilea steag este repartizat în mod similar, se alege de pe fiecare rînd cel mai înalt elev, iar dintre cei aleși, cel mai scund. În cazul în care există mai mulți elevi cu aceeași înălțime, se alege primul dintre ei. Să se scrie un program care să afișeze înălțimile purtătorilor de steag; valorile  $m$ ,  $n$  și înălțimile elevilor se citesc de la tastatură.

Ca indicație, concurenților li s-a prezentat următorul exemplu ( $m=3$ ,  $n=4$ ):

rîndul 1:	120	130	140	150
rîndul 2:	110	120	120	130
rîndul 3:	140	140	150	150

Primul steag este dat elevului din poziția (3,1), cu înălțimea 140, iar al doilea steag elevului din poziția (2,1), cu înălțimea 130.

#### Clasa a VI-a

O școală, are maximum 7 serii de clasa a VI-a, notate cu A, B, C, D, E, F, G. Să se scrie un program care să genereze o planificare a întâlnirilor sportive între aceste clase, astfel încît fiecare să se întâlnească o singură dată cu altă clasă.

Întîlnirile au loc zilnic, cîte una în fiecare zi, cu excepția zilelor de duminică. Programarea întâlnirilor sportive începe cu ziua de miercuri 1 iunie 1988 și arată astfel:

miercuri 1 iunie 1988; VI A - VI B  
joi 2 iunie 1988; VI A - VI C

#### Clasa a VII-a

Elevii unei clase sînt specificați prin: nume, nota la română, nota la matematică, nota la fizică și nota la chimie. Să se scrie un program care întîi citește datele despre elevi și apoi afișează, pentru fiecare materie în parte, procente notelor de 5-6, 7-8, 9-10 și nota cea mai frecventă și de cîte ori apare. În final se cere și nota cea mai frecventă obținută de elevii clasei la toate materiile.

#### Clasa a VIII-a

Un grup de elevi, identificați prin numerele 1, 2, ...,  $n$ , sînt așezați într-un cerc în ordinea 1, 2, ...,  $n$ .

Începînd cu elevul următor lui  $k < n$ , ales aleator, se numără de la 1 la  $m$ , eliminînd din cerc pe cel la care s-a oprit numărătoarea. Procedul se repetă începînd cu elevul următor pînă cînd rămîne unul singur. Să se scrie un program care tipărește ordinea de eliminare a elevilor, valorile  $n$  și  $m$  se citesc de la tastatură.

Concurenților li s-a prezentat următorul exemplu:

$n=8$ ,  $m=2$ ,  $k=4$ , ordinea inițială = (1 2 3 4 5 6 7 8), ordinea eliminării = (6 8 2 4 7 3 1 5).



## REZOLVĂRI

### Soluția problemei de la clasa a V-a 1988

```

10 input "numărul rândurilor="; m
15 if m < 1 or m > 15 then goto 10
20 input "numărul coloanelor="; n
25 if n < 1 or n > 6 then goto 20
40 let x=0:let y=999
50 for i=1 to m
60   let b=999:let c=0
70   for j=1 to n
80     input "Înălțimea elevului din rândul "; (i); " și coloana "; (j); "="; a
90     print at i+2, (j-1)*5; a
100    if a < b then let b=a
110    if a > c then let c=a
120    next j
130    if b > x then let x=b
140    if c < y then let y=c
150  next i
160 print at 18,0; "Primul stegar are înălțimea "; x
170 print at 19,0; "Al doilea stegar are înălțimea "; y

```

### Soluția problemei de la clasa a VI-a 1988

```

10 INPUT "nr. clase="; nc
15 DIM c$(nc):
  DIM z$(7, 9)
16 FOR i=1 TO 7:
  READ z$(i):
  NEXT i
20 FOR i=1 TO nc:
  READ c$(i):
  NEXT i
25 LET dat=0
  LET zi=0
30 FOR i=1 TO nc-1
40   FOR j=i+1 TO nc
50     LET dat=dat+1:
     LET zi=zi+1
60     IF zi=5 THEN PRINT z$(zi); dat; TAB 12; "iunie 1988":
     LET dat=dat+1:
     LET zi=zi+1
70     PRINT z$(zi); dat; TAB 12; "iunie 1988 VI"; c$(i); "-VI"; c$(j)
100    IF zi=7 THEN LET zi=0
110  NEXT j
120 NEXT i
199 DATA "Miercuri", "joi", "Vineri", "Sâmbătă", "Duminică", "Luni", "Marți"
200 DATA "A", "B", "C", "D", "E", "F", "G"

```

### Soluția problemei de la clasa a VII-a 1988

```

10 READ ne
20 DIM e$(ne, 10):
  DIM n(ne, 4):
  DIM f(5, 6):
  DIM d$(4, 10)
30 FOR i=1 TO 4:
  READ d$(i):
  NEXT i
40 PRINT "Nume rom mat fiz chim"
50 FOR i=1 TO ne
60   READ e$(i):
   PRINT e$(i);

```

## 4. PROBLEME PROPUSE ȘI REZOLVATE (V. 16.5)



```

70 FOR j=1 TO 4
80 READ n(i, j):
PRINT n(i, j); " ";
90 NEXT j
100 PRINT
110 NEXT i
120 FOR i=1 TO 5:
FOR j=1 TO 6
130 LET f(i, j)=0
140 NEXT j
NEXT i
150 FOR i=1 TO ne:
FOR j=1 TO 4
160 LET f(j, n(i, j)-4)=f(j, n(i, j)-4)+1
170 NEXT j:
NEXT i
180 FOR i=1 TO 4:
FOR j=1 TO 6
190 LET f(5, j)=f(5, j)+f(i, j)
200 NEXT j
NEXT i
210 FOR i=1 TO 4
220 PRINT 'd$(i)
230 GO SUB 360
240 PRINT "Note de 5-6: "; p56; "%"
250 PRINT "Note de 7-8: "; p78; "%"
260 PRINT "Note de 9-10: "; p910; "%"
270
280 GO SUB 400
290 PRINT "nota"; k+4; "apare de"; max; "ori"
300 NEXT i
310 LET i=5:
GO SUB 400
320 PRINT
330 PRINT "Nota cu frec. maxima este "; k+4
340 PRINT "Ea apare de "; max; "ori"
350 STOP
360 LET p56=(f(i, 1)+f(i, 2))/ne*100
370 LET p78=(f(i, 3)+f(i, 4))/ne*100
380 LET p910=(f(i, 5)+f(i, 6))/ne*100
390 RETURN
400 LET max=f(i, 1):
LET k=1
410 FOR j=2 TO 6
420 IF max<=f(i, j) THEN LET max=f(i, j):
LET k=j
430 NEXT j
440 RETURN
450 PRINT "Nume rom mat fiz chim"
460 DATA 4
470 DATA "Romana", "Matematica", "Fizica", "Chimie"
480 DATA "Popescu", 8, 6, 7, 10
490 DATA "Ionescu", 6, 7, 8, 9
500 DATA "Petrescu", 8, 9, 9, 10
510 DATA "Marinescu", 8, 9, 9, 10

```

*Solufia problemei de la clasa a VIII-a 1988*

```

10 INPUT "nr elevi="; n
20 PRINT "nr elevi="; n
30 DIM c(n)
40 INPUT "valoare de numarare="; m
50 PRINT "valoare de numarare="; m

```



```

60 FOR i=1 TO n:
    LET c(i)=i
    NEXT i
70 PRINT „Ordinea inițială a elevilor este“
80 LET ne=n
90 LET i=INT ((n-1)*RND)+1
100 PRINT „Elevi={“;
110 FOR j=1 TO n-1
120 IF j=i THEN PRINT INVERSE 1; c(j); INVERSE 0; “;
    GO TO 140
130 PRINT C(j); “;”
140 NEXT j
150 PRINT C(n); “}”
160 PRINT „Ordinea de eliminare a elevilor este“
170 PRINT “Elevi={“
180 LET ia=0
190 LET na=m-ne*INT (m/ne)
200 LET i=i+1
210 IF i>n THEN LET i=1
220 IF c(i)=0 THEN GO TO 200
230 LET ia=ia+1
240 IF na=0 THEN LET na=ne
250 IF ia<>na THEN GO TO 200
260 PRINT c(i); “;”
270 LET c(i)=0
280 LET ne=ne-1
290 IF ne<=1 THEN GO TO 310
300 GO TO 180
310 FOR i=1 TO n
320 IF c(i)<>0 THEN PRINT c(i); “}”:
    GO TO 340
330 NEXT i
340 STOP

```

#### Soluția optimă a problemei de la clasa a VIII-a 1988

```

10 input "n="; n
15 if n<1 or n<>int(n) then goto 10
20 input "m="; m
25 if m<1 or m<>int(m) then goto 20
30 dim a(n)
40 for i=1 to n
50 let a(i)=i-int(i/n)*n+1
60 next i
70 let k=int(1+rnd*n)
80 for i=1 to (m-1)-int((m-1)/n)*n
90 let k=a(k)
100 next i
110 print a(k)
120 let a(k)=a(a(k))
130 let n=n-1
140 if n>0 then goto 80

```

### 16.5.3. Concursul național de informatică 10–18 iulie 1989

#### PROBLEME PROPUSE

##### Clasa a V-a

Se generează aleator trei numere naturale mai mici decât 100 și diferite de zero. Să se afle suma inverselor lor, adică  $a/b=1/x+1/y+1/z$ , unde  $a/b$  este fracție ireductibilă.

##### Clasa a VI-a

Să se calculeze anul, ziua, luna și ora revenirii unei rachete pe pământ cunoscându-se anul, ziua, luna și ora plecării și durata zborului în minute. Zborul durează cel mult un an.

#### 4. PROBLEME PROPUSE ȘI REZOLVATE (V. 16.5)



## REZOLVĂRI

### Clasa a VII-a

Se dă un vector de numere naturale mai mici decât 101 și un număr natural, A. Să se insereze între două componente vecine, a căror diferență în valoare absolută este mai mare sau egală cu A, media aritmetică a lor, pentru ca în final să rezulte un vector în care diferența absolută dintre două elemente vecine este mai mică decât A.

### Clasa a VIII-a

Se dau trei perechi de numere naturale  $(x_1, y_1)$ ,  $(x_2, y_2)$ ,  $(x_3, y_3)$  care reprezintă coordonatele a trei puncte în plan, unde  $x, y$  aparțin  $[1, 150]$ . Să se scrie un program care verifică dacă acestea pot forma un triunghi și în caz afirmativ să se deseneze și calculeze suprafața acelui triunghi și natura sa: isoscel, echilateral, dreptunghic, oarecare.

### Soluția problemei de la clasa a V-a 1989

```

10 LET X=INT (RND*100)+1
20 LET Y=INT(RND*100)+1
30 LET Z=INT(RND*100)+1
40 LET A=Y*Z+X*Z+X*Y
50 LET B=X*Y*Z
60 LET C=A
70 LET D=B
80 LET R=C-INT(C/D)*D
90 IF R=0 THEN LET CMMDC=D:GOTO 110
100 LET C=D:LET D=R:GOTO 80
110 LET A=A/CMMDC:LET B=B/CMMDC
120 PRINT 1; "/"; X; "+"; 1; "/"; Y; "+"; 1; "/"; Z; "="; A; "/"; B
    
```

### Soluția problemei de la clasa a VI-a 1989

```

10 DIM I(12)
20 FOR i=1 TO 12:
    READ I(i):
    NEXT i
30 DATA 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
40 PRINT "Introduceți data curentă:"
50 INPUT "an="; an:
    PRINT "an="; an;
60 INPUT "luna="; luna
70 IF luna <1 OR luna >12 THEN GO TO 60
80 PRINT "luna="; luna;
90 LET x=an
100 IF luna >2 THEN LET x=x+1
110 IF x=4*INT (x/4) AND x<>100*INT (x/100) OR x=400*INT (x/400) THEN LET
    I(2)=I(2)+1
120 INPUT "zi="; zi
130 IF zi <1 OR zi >1 (luna) THEN GO TO 120
140 PRINT "zi="; zi;
150 INPUT "ora="; ora
160 IF ora <0 OR ora >23 THEN GO TO 150
170 PRINT "ora="; ora;
180 INPUT "minut="; min
190 IF min <0 OR min >59 THEN GO TO 180
200 PRINT "min="; min
210 INPUT "Introduceți durata (în minute)="; d
220 PRINT "Durata (în minute)="; d
240 LET d=d+min
250 LET min=d-60*INT (d/60)
260 LET d=INT (d/60)+ora
270 LET ora=d-24*INT (d/24)
280 LET d=INT (d/24)+zi
    
```



```

290 IF d<=1 (luna) THEN LET zi=d:
    GO TO 330
300 LET d=d-1 (luna)
310 IF luna=12 THEN LET luna=1:
    LET an=an+1:
    GO TO 290
320 LET luna=luna+1:
    GO TO 290
330 PRINT "an="; an; "luna="; luna; "zi="; zi; "ora="; ora; "min="; min

```

*Soluția problemei de la clasa a VII-a 1989*

```

10 CLS
20 INPUT "Număr de elemente<101="; N
30 IF N<2 OR N>100 THEN GOTO 20
40 DIM V(100)
50 INPUT "a="; A
60 PRINT : PRINT "a="; A
70 IF A<2 THEN GOTO 50
80 FOR I=1 TO N
90 INPUT "Elementul <101="; V(I)
100 IF V(I)>100 OR V(I)<0 THEN GOTO 90
110 PRINT "v("; I; ")="; V(I)
120 NEXT I
130 FOR I=1 TO N-1
140 IF ABS(V(I)-V(I+1))<A THEN GOTO 200
150 FOR J=N TO I+1 STEP -1
160 LET V(J+1)=V(J)
170 NEXT J
180 LET N=N+1
190 LET V(I+1)=INT((V(I)+V(I+2))/2)
200 NEXT I
210 FOR I=1 TO N-1
220 IF ABS(V(I)-V(I+1))>=A THEN GOTO 130
230 NEXT I
240 FOR I=1 TO N
250 PRINT V(I); " ";
260 NEXT I

```

*Soluția problemei de la clasa a VIII-a 1989*

```

10 DIM x(3):
    DIM y(3)
20 LET eps=0.001
30 PRINT "Introduceți coordonatele:"
40 FOR i=1 TO 3
50 INPUT "x("; i; ")="; x(i); "y("; i; ")="; y(i)
60 PRINT "x("; i; ")="; x(i); "y("; i; ")="; y(i)
70 PLOT x(i), y(i)
80 NEXT i
90 REM desen
100 PLOT x(1), y(1)
110 DRAW x(2)-x(1), y(2)-y(1)
120 DRAW x(3)-x(2), y(3)-y(2)
130 DRAW x(1)-x(3), y(1)-y(3)
140 REM Calcul arie
150 LET arie=ABS ((x(2)*y(3)+x(1)*y(2)+x(3)*y(1)-x(2)*y(1)-x(3)*y(2)-x(1)*y(3))/2)
160 IF arie<=eps THEN PRINT "Triunghiul nu exista"
    STOP
170 PRINT "Aria="; arie
180 REM Calcul lungimi laturi

```



```

190 LET a = SQR ((x(2)-x(1))*(x(2)-x(1))+(y(2)-y(1))*(y(2)-y(1)))
200 LET b = SQR ((x(3)-x(2))*(x(3)-x(2))+(y(3)-y(2))*(y(3)-y(2)))
210 LET c = SQR ((x(1)-x(3))*(x(1)-x(3))+(y(1)-y(3))*(y(1)-y(3)))
220 LET oarecare = 1
230 IF ABS (a-b) < eps AND ABS (a-c) < eps THEN PRINT "triunghi echilateral"
    LET oarecare = 0
240 IF ABS (a*a-b*b-c*c) < eps OR ABS (b*b-a*a-c*c) < eps OR ABS (c*c-a*a-b*b) < eps THEN PRINT "Triunghi dreptunghic"
    LET oarecare = 0
250 IF ABS (a-b) < eps OR ABS (a-c) < eps OR ABS (b-c) < eps THEN PRINT "Triunghi is oscar"
    LET oarecare = 0
260 IF oarecare = 1 THEN PRINT "Triunghi oarecare"

```

## 5. PROBLEME REZOLVATE ÎN PSEUDOCOD

\*)

1. Știind că  $p\%$  din producția zilnică a unui muncitor este evaluată la  $a$  lei, să se determine care este suma realizată dacă își îndeplinește norma sută la sută.

Notind cu  $s$  suma cerută (a și  $p$  fiind informațiile de intrare, iar  $s$  informația de ieșire), rezultă soluția:

```

read a, p
s := (100 . a) / p
write s

```

2. Fie  $x$  o variabilă numerică. Să se determine valoarea ei absolută.

Soluție.

```

read x
if x >= 0
then
    a := x
else
    a := -x
endif
write a

```

3. Fie  $a, b, c$  numere oarecare. Să se calculeze aria și perimetrul triunghiului cu aceste laturi (dacă  $a, b, c$  pot fi laturile unui triunghi) sau să se precizeze dacă nu pot determina un triunghi.

Soluție. Observăm că dacă cea mai mare dintre cele trei valori  $a, b, c$  (fie  $m$  aceasta) este mai mică decât suma celorlalte două, atunci  $a, b, c$  pot constitui laturile unui triunghi. Ca atare vom determina pe  $m$  după care condiția

$$m < a + b + c - m$$

va decide dacă  $a, b, c$  sînt sau nu laturi de triunghi, după cum este, respectiv, nu este ea îndeplinită ( $a + b + c - m$ ) reprezintă suma celor două valori mai mici deoarece  $m$  provine sau din  $a$ , sau din  $b$ , sau din  $c$ ).

Notind cu  $s$  aria, cu  $q$  perimetrul și cu  $p$  semiperimetrul, răspunsul este:

```

read a, b, c
m := a

```

```

if b > m
then
    m := b
endif
if c > m
then
    m := c
endif
if d > m
then
    m := d
endif
if m < a + b + c - m
then
    q := a + b + c
    p := q / 2
    s := p . (p - a) . (p - b) . (p - c)
(1/2)
write s, q
else
    write „Nu formează triunghi”
endif

```

(unde cu  $\uparrow$  s-a notat ridicarea la putere (de exemplu  $a \uparrow (1/2)$  înseamnă  $\sqrt{a}$ ).

4. Fie  $a$  un număr natural în baza 10. Să se determine numărul cifrelor corespundentului său binar și să se precizeze cîte dintre cifre sînt 1.

Soluție. Notăm cu  $c$  numărul cifrelor corespundentului binar al lui  $a$ , cu  $u$  cîte dintre cifre sînt 1, iar cu  $q$ , partea întregă a lui  $x/2$ . Cu aceste notații soluția ce se propune este următoarea:

```

read a
e := 0
u := 0
while x > 1 do
    q := x / 2
    r := x - 2 * q
    c := c + 1
    if r = 1
    then
        u := u + 1
    end if

```

\* Articol „Aplicații ale noțiunilor de informatică studiate la cap. „ALGORITHMI” — clasa a IX-a; de Cercet. șt. dr. Stelian Niculescu și prof. Gh. N. Rîzescu, în: Revista ÎNVĂȚĂMÎNTUL LICEAL și TEHNIC PROFESIONAL din Decembrie 1987.



```

endif
x := q
endwhile
write c, u

```

Menționăm că variabila  $x$  este auxiliară (de lucru), ea fiind folosită pentru a conserva valoarea lui  $a$ , iar  $r$  reprezintă restul împărțirii lui  $x$  la 2.

5. Să se rezolve ecuația  $a \cdot x + b = 0$

**Soluție:**

```

read a, b
if a = 0
then
x := -b/a
write x
else
if b = 0
then
write „Nedeterminare“
else
write „Imposibilitate“
endif
endif

```

6. Să se determine

$m = \max(a, b, c)$

**Soluție.** Urmărind calea similară celei din exemplul 4, rezultă:

```

read a, b, c, d
m := a
if b > m
then
m := b
endif
if c > m
then
m := c
endif
if d > m
then
m := d
endif
write m

```

7. Să se genereze primii  $n$  termeni ai șirului: 1, 2, 4, 8, 16 ...

**Soluție.** Observînd că este vorba de puterile naturale ale lui 2, rezultă soluția:

```

read n
i := -1
c := 1
while i ≤ n do
write c
c := c.2
i := i + 1
endwhile

```

8. Să se găsească al  $n$ -lea termen din șirul lui Fibonacci ( $n > 2$ ):  
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

**Soluție.** Se remarcă faptul că un termen al șirului este suma celor doi care-l preced imediat (primii doi termeni ai șirului fiind dați). De aceea să notăm cu  $a, b, c$  trei termeni consecutivi. Pentru început  $a=0$ ,  $b=1$  din care rezultă  $c=a+b=1$ . Făcînd  $a=b$  (deci  $a=1$ ),  $b=c$  (deci  $b=1$ ), rezultă  $c=a+b=2$ . Se repetă aceste operații atîta timp cît nu s-a generat al  $n$ -lea termen. Pentru evidențierea generării celui de-al  $n$ -lea termen s-a destinat variabila  $k1$  a cărei valoare de început este 2. Deci:

```

read m
a := 0
b := 1
k := 2
while k ≤ m do
c := a + b
a := b
b := c
k := k + 1
endwhile
write c

```

9. Se consideră  $n$  valori precizate una cîte una. Să se spună cîte dintre ele sînt zero.

**Soluție.**

```

read n
x := 1
m := 0
while x ≤ n do
read v
if v = 0
then
m := m + 1
endif
x := x + 1
endwhile
write m

```

unde  $m$  este variabila destinată numărării zerourilor,  $x$  variabila care ține evidența valorilor analizate, iar  $v$  valoarea analizată.

10. Dîndu-se un număr natural  $n$ , să se precizeze dacă este prim.

**Soluție.** Se știe că un număr este prim dacă nu se divide cu numerele prime mai mici decît  $\sqrt{n}$ . Pentru a evita construirea tuturor numerelor prime mai mici ca  $\sqrt{n}$ , vom considera că  $n$  este prim dacă nu este par și nu se divide cu numerele impare mai mici decît  $[n/2]$  (mulțimea numerelor impare mai mici ca  $[n/2]$  incluzînd mulțimea numerelor prime mai mici ca  $\sqrt{n}$ ). Vom utiliza variabila  $p$  a cărei valoare se definește astfel:

$$p = \begin{cases} 0, & \text{dacă } n \text{ este prim} \\ 1, & \text{dacă } n \text{ nu este prim} \end{cases}$$

În aceste condiții soluția este:

```

read n

```



```

if n=[n/2] . 2
then
  write „n nu este prim“
else
  p := 0
  x := [n/2]
  i := 3
  while i ≤ x do
    if n=[n/i] . i
    then
      p := 1
      i := x
    endif
    i := i+2
  endwhile
  if p=1
  then
    write „n nu este prim“
  else
    write „n este prim“
  endif
endif

```

În cadrul structurii while/do este prezentată pseudostructura alternativă if/then care evidențiază apariția unui divisor i al lui n (prin  $p := 1$ ) și în același timp impune lui i să ia valoarea finală x (prin  $i := x$ ) pentru a se ieși din while/do.



## 6. SOLUȚII ȘI INDICAȚII SUPLIMENTARE DE REZOLVARE A TEMELOR RECOMANDATE ÎN SUBCAPITOLUL 16.6

### Tema nr. 1

Imaginea 3D a „pălăriei de mexican“ se obține pe baza formulei date cu ajutorul programului:

```
100 FOR x=40 TO 215
110 LET b=999: LET t=0
120 FOR y=16 TO 144 STEP 4
130 LET r=SQR ((x-127) * (x-127)+(y-80) * (y-80)) /15
140 LET z=INT (y+90 * EXP (-r/3) * Cos r)
150 IF z < b OR z > t THEN PLOT x, z
160 IF z < b THEN LET b=z
170 IF z > t THEN LET t=z
180 NEXT y
190 NEXT x
```

100—190 Ciclează cu x luind valori de la 40 la 215.

110 b și t sînt folosite pentru a elimina liniile ascunse; aici ele sînt inițializate cu valori în afara intervalului în care se află de obicei.

120—180 Ciclează cu y luind valori de la 16 la 144 din 4 în 4.

130—140 Calculează înălțimea de desenare a punctului.

150 Desenează punctul, dacă nu este ascuns după o porțiune a funcției din prim plan.

160—170 Actualizează b și t cu cea mai mică, respectiv cea mai mare ordonată, din cadrul punctelor desenate pînă acum pentru coordonata x.

### Tema nr. 2

Se va aplica metoda backtracking de elaborare a algoritmilor, care în acest caz particular constă în alegerea poziției celor 8 dame, așezînd succesiv piesele pe tabla de șah, în așa fel încît să fie verificată condiția problemei (adică damele așezate trebuie „să nu se ia“ una pe alta). Concret, presupunînd că pe tabla de șah se află deja așezate  $(k-1)$  dame care „nu se iau“ una pe alta, alegem pentru următoarea damă  $(k)$  prima poziție pentru care condiția problemei rămîne valabilă.

Dacă nu există nici o poziție care să satisfacă condiția dată, revenim la pasul anterior încercînd să reasezăm dama  $(k-1)$  în următoarea poziție, în care condiția rămîne adevărată pentru primele  $(k-1)$  dame.

Dacă există o asemenea poziție pentru piesa  $(k)$ , atunci se continuă algoritmul de așezare cu piesa  $(k+1)$ .

Algoritmul se încheie (după ce s-au generat toate soluțiile) cînd nu mai putem reaseza prima damă într-o nouă poziție.

### Tema nr. 3

Trasarea elipsei se poate face folosind programul de la punctul 18.20 sau utilizînd formula parametrizată:

$$x = a \cos d$$

$$y = b \sin d$$

unde parametrul  $d$  aparține intervalului  $(0, 2\pi)$ , iar  $a$  și  $b$  sînt semiaxele elipsei.

Aria elipsei se calculează cu formula:

$$A = 2\pi ab$$



## Tema nr. 5

Programul\* FUNCȚIA DE GRADUL II CU UN PARAMETRU REAL a fost conceput în scopul folosirii sale de către profesori și elevi în cadrul orelor de matematică în care se studiază familii de funcții de gradul II.

Trasarea graficelor cu ajutorul calculatorului prezintă următoarele avantaje: precizie sporită, posibilitatea trasării graficelor la orice scară, calcularea rapidă a unor caracteristici.

După încărcarea programului este afișat un meniu cuprinzând următoarele opțiuni:

- 1 — Exemple
- 2 — Grafice
- 3 — Probleme

### Exemple

Alegerea opțiunii 1 duce la afișarea unui alt meniu care permite alegerea unor familii de funcții cu diferite caracteristici (graficele nu au nici un punct comun, graficele au unul sau două puncte comune, virfurile graficelor sînt pe o dreaptă sau sînt pe o parabolă). În urma alegerii uneia din opțiunile afișate se trasează 6 grafice din familia respectivă de funcții și sînt afișate coordonatele punctelor comune sau ecuația dreptei ce unește virfurile parabolilor dacă este cazul.

În timpul trasării graficelor, pe ultima linie a ecranului, sînt afișate coordonatele virfului și valoarea parametrului  $m$ .

### Grafice

În cazul în care se dorește trasarea graficelor unei familii de funcții introduse de utilizator se alege opțiunea 2 și apoi se introduc coeficienții  $A$ ,  $B$ ,  $C$  ca funcții de  $m$ .

Se introduc valorile între care variază parametrul  $m$  și parametrul  $t$  în funcție de care se alege scara la care se trasează graficele.

După ce se trasează 6 grafice, sînt afișate atunci cînd este cazul, coordonatele punctelor comune și ecuația dreptei sau parabolei care unește virfurile graficelor.

### Probleme

În cadrul opțiunii 3 sînt afișate enunțurile și rezolvările a 3 probleme precum și graficele funcțiilor.

Sînt propuse spre rezolvare enunțurile a 3 probleme.

În cazul în care apare o eroare sau a fost oprită execuția programului, acesta se poate relansa cu GO TO 5000.

Programul poate fi îmbunătățit sau modificat în scopul realizării unui program care să cuprindă alte proprietăți ale funcției de gradul II.

## Tema nr. 13

În categoria programelor-instrumente de preț la îndemîna profesorilor de matematică, se constituie și programul „LOCURI GEOM”, care ilustrează un mod de rezolvare a temei și a unor probleme identice sau similare celor propuse.

Programul tratează un set de 6 probleme din manualul de geometrie de clasa a IX-a; rezolvarea acestor probleme cu ajutorul calculatorului este nu numai indicată, dar și absolut necesară pentru învățămîntul modern. Problemele rezolvate în acest mod sînt din categoria celor de locuri geometrice. Întuirea formei locului geometric cerut în enunțul problemei este foarte dificilă dacă s-ar folosi mijloace de învățămînt tradiționale. Aportul calculatorului la acest domeniu al geometriei este foarte eficient ținînd seama de faptul că prin program se

---

\* Programele FUNCȚIA DE GRADUL II și LOCURI GEOMETRICE sînt elaborate de elevii Mărgineanșu Dragoș, Kleitsch Călin, respectiv prof. Dorin Mani și prof. Dorel Mihet de la Liceul de matematică-fizică nr. 1 din Timișoara, fiind incluse în volumul INSTRUIRE ASISTATĂ DE CALCULATOR editat de ITCI în 1989, sub coordonarea Nicolae Badea, Ioan Jurcă și Stelian Niculescu.



calculează efectiv poziția fiecărui punct care aparține locului geometric. Calculele se bazează pe elemente de geometrie analitică. Pentru facilitarea întuirii formei locului geometric înainte de generarea completă a locului geometric respectiv, se calculează câteva puncte izolate care aparțin locului și se trasează pe ecran. După afișarea fiecărui punct se trece la generarea următorului punct. După calculul citorva puncte programul se derulează fără intervenția utilizatorului până la afișarea completă a locului geometric respectiv. În final după apăsarea unei taste se afișează pe ecran un text care explicitează forma locului geometric cerut de problema respectivă. Locul geometric care rezultă prin calcule se generează pe ecran cu altă culoare pentru a se deduce mai ușor și vizual forma.

Dăm în cele ce urmează textul problemelor rezolvate:

1. Fie  $AB$  un diametru fix al unui cerc  $C(O, r)$ , iar  $M$  un punct variabil pe cerc. Se ia pe raza  $OM$  un punct  $P$  astfel încât  $OP = d(M, AB)$ . Să se afle locul geometric al punctului  $P$ .
2. Fie  $AB$  o coardă fixă a unui cerc, iar  $PO$  o coardă variabilă ca poziție, dar de lungime fixă. Să se afle locul geometric al punctelor:  $AP$  intersectat cu  $BO$  și  $AC$  intersectat cu  $BP$ .
3.  $AB$  fiind o coardă fixă iar  $M$  un punct variabil al unui cerc, să se afle locul geometric al punctului  $P$ , astfel încât  $M$  aparține lui  $AP$  și  $MP = MB$ .
4. Se consideră un cerc, o coardă fixă  $AB$  și un punct variabil pe cerc. Să se determine locul geometric al ortocentrului triunghiului  $AMB$ .
5. Se consideră un cerc, o coardă fixă  $AB$  și un punct  $M$  variabil pe cerc. Să se determine locul geometric al centrului de greutate al triunghiului  $AMB$ .
6. Fie  $A$  un punct fix iar  $P$  un punct variabil al unui cerc. Să se afle locul geometric al punctului  $M$  de intersecție a bisectoarei unghiului  $FOA$  cu cercul circumscris triunghiului  $POA$ .

Programul nu are un caracter interactiv cu excepția rezolvării problemei a doua când se cere utilizatorului să aleagă între două mărimi pentru lungimea coardei variabile. Acest lucru este determinat de dificultatea încadrării în ecran pentru vizualizarea soluției. La mărimi alese arbitrar ar putea exista posibilitatea neapariției figurii sau locului pe ecran cu toate că programul lucrează. Există posibilitatea reîntoarcerii la oricare problemă pe baza unui meniu care se afișează la terminarea rezolvării fiecărei probleme. De asemenea, la revenirea într-o problemă se afișează la început textul problemei asemenea unei mașini de scris după care la apăsarea unei taste se derulează construirea locului geometric cerut prin problema respectivă.

Programul aparține unui ciclu de programe pentru locuri geometrice care rezolvă în întregime acest domeniu din geometria clasei a IX-a.







```

410 GO TO 480
420 LET x2=(b1-b)/(2*(a-a1)): LET x1=x2
430 PRINT " Graficele au un punct comun: "
440 LET x=x1
450 PRINT " A("x";";"VAL f$;"")"
460 GO TO 480
470 LET swcirc=1: PRINT " Graficele nu au puncte comune"
480 IF w(1,1)-w(1,2)=0 THEN GO TO 750
485 LET i=(w(2,1)-w(2,2))/(w(1,1)-w(1,2)): LET j=w(2,1)-i*(1,1)
490 FOR f=1 TO 6: IF w(1,f)=0 AND w(2,f)=0 THEN GO TO 510
500 IF i*w(1,f)+j*w(2,f)-.08 OR i*w(1,f)+j*w(2,f)+.08 THEN GO TO 540
503 NEXT f
510 LET i=(INT (i*100))/100: LET j=(INT (j*100))/100
520 PRINT " Virfurile parabolilor sint pe o dreapta a carei ecuatie este:
("i";"x+("j";"y)
530 LET h$="i*x+j": PAUSE 0: GO TO 660
540 IF w(1,1)*w(1,1)-w(1,2)*w(1,2)+w(1,1)*w(1,1)-w(1,3)*w(1,3))*w(1,1)-w(1,2)
)=0 THEN GO TO 750
542 LET q=((w(1,1)*w(1,1)-w(1,2)*w(1,2))*(w(1,1)-w(1,3)))-((w(1,1)*w(1,1)-w(1,
3)*w(1,3)*(w(1,1)-w(1,2)))
543 LET s=((w(1,1)-w(1,3))*(w(2,1)-w(2,2)))-((w(1,1)-w(1,2))*(w(2,1)-w(2,3)))
544 LET i=s/q
545 IF w(1,1)-w(1,2)=0 THEN GO TO 750
547 LET j=(w(2,1)-w(2,2)-i*(w(1,1)-w(1,2)*w(1,2)))/(w(1,1)-w(1,2))
553 LET k=w(2,1)-i*w(1,5)*w(1,1)-j*w(1,1)
570 FOR i=1 TO 6: IF w(1,f)=0 THEN GO TO 590
580 IF i=w(1,f)*w(1,f)+j*w(1,f)+k*w(2,f)-.25 OR i=w(1,f)*w(1,f)+j*w(1,f)+k*w(2,
f)+.25 THEN GO TO 750
590 NEXT f: LET i=(INT (i+10))/10: LET j=(INT (j+10))/10: LET k=(INT (k+10))/10
600 PRINT " Virfurile parabolilor sint pe o parabola a carei ecuatie este:
("i";"x +("j";"x+("k";"y"
610 LET h$="i*x*x+j*x+k"
620 PAUSE 0: GO TO 660
630 IF INKEY$="D" OR INKEY$="d" THEN STOP
640 IF INKEY$="n" OR INKEY$="N" THEN PAUSE 0: CLS : GO TO 5000
650 GO TO 630
660 RANDOMIZE USR 50112
670 IF swcirc=1 THEN GO TO 700
680 LET x=x1: PLOT INK 2;x+127,87+VAL f$
690 LET x=x2: PLOT INK 2;x+127,87+VAL f$
700 FOR x=-127/t TO 127/t STEP 1/t
710 LET y=VAL h$
720 IF y*t<-87 OR y*t>87 THEN GO TO 740
730 PLOT INK 2: x*t+127,y*t+87
740 NEXT x: PAUSE 0: GO TO 280
750 RANDOMIZE USR 50112
780 GO TO 280
785 REM INCEPUT.1dir: def UDG.Meniu
786 FOR f=50100 TO 50123: READ b: POKE f,b: NEXT f
789 DATA 17,0,167,33,0,64,1,0,27,237,176,201,17,0,64,33,0,167,1,0,27,237,176,20
1
799 CLS : FOR f=0 TO 55: read b: POKE 65368+f,b: NEXT f: RESTORE
800 DATA BIN 1110000,BIN 10010000,BIN 1000000,BIN 10000000,BIN 11111000,0,0,0,0,0
,0,0,0,BIN 11111000,BIN 10101000,BIN 10101000

```



```

803 DATA BIN 10000,BIN 110000,BIN 1110000,BIN 11111000,BIN 11001100,BIN 1000011
0,BIN 11,BIN 11,BIN 11,BIN 11,BIN 10000110,BIN 11001100,BIN 11111000,BIN 1110000
,BIN 110000,BIN 10000
805 DATA BIN 1,BIN 10,BIN 100,BIN 1111111,BIN 10000,BIN 11111111,BIN 1000000,B
IN 10000000,0,BIN 11110,BIN 110000,BIN 1000000,BIN 1111110,BIN 1000000,BIN 11000
0,BIN 11110
806 DATA 0,0,BIN 1100110,BIN 10011001,BIN 10011001,BIN 10010110,BIN 1100000,0
5000 PAPER 6: INK 1: BORDER 4: CLS
5005 PRINT AT 9,1;"OPTIUNI"
5010 PRINT AT 5,1;"1 - Exemple"
5015 PRINT AT 7,1;"2 - Grafice"
5017 PRINT AT 9,1;"3 - Probleme"
5020 IF INKEY$="3" THEN GO TO 7000
5030 IF INKEY$="2" THEN GO TO 5500
5040 IF INKEY$="1" THEN GO TO 6000
5050 GO TO 5020
5500 GO TO 200
6000 CLS : PRINT AT 2,1;"1 - Graficele nu au puncte comune";AT 5,1;"2 -
Graficele au un punct comun";AT 7,1;"3 - Graficele au doua puncte comune
";AT 10,1;"4 - Graficele au virfurile pe o dreapta ";AT 13,1;"5 - Graficele
au virfurile pe o parabola"
6005 PRINT AT 16,1;"ENTER - Meniu"
6010 IF INKEY$="1" THEN GO TO 6500
6020 IF INKEY$="2" THEN GO TO 6600
6030 IF INKEY$="3" THEN GO TO 6700
6040 IF INKEY$="4" THEN GO TO 6800
6050 IF INKEY$="5" THEN GO TO 6900
6055 IF INKEY$=CHR$ 13 THEN GO TO 5000
6080 GO TO 6010
6500 LET m$="m": LET n$="2": LET p$="2+m": GO TO 6605
6600 LET m$="m*(m-1)": LET n$="m-2": LET p$="3"
6605 LET t=3
6606 LET f$="("+m$+" ) * x + (" + n$ + " ) * x + p$ : CLS : PRINT AT 8,0; " A=";m$: PRINT "
B=";n$: PRINT " C=";p$: LET v=-0.2: LET w=.2: PRINT AT 12,3;"f (x)=";m$;"x +"
;n$;"x"+p$: LET sw=0: PAUSE 0: GO TO 10
6700 LET m$="m": LET n$="m+1": LET p$="-1-6*m": LET t=3: LET v=-.2: LET w=.2: PA
USE 0: GO TO 6605
6800 LET m$="m": LET n$="2*(m-1)": LET p$="m-1": GO TO 6605
6900 LET m$="(m+1)/2": LET n$="2*m": LET p$="(m-1)/(m+1)": LET t=10
6905 GO TO 6605
7000 CLS
7010 PRINT AT 5,1;"1 - Rezolvate"
7012 PRINT AT 7,1;"2 - Propuse spre rezolvare"
7015 PRINT AT 9,1;"ENTER - Meniu"
630 570>FOR f=1 TO 6:IF v(1,f)=0 AND v(2,f)=0 THEN GO TO 590
580 IF i*w(1,f)*w(1,f)+j*w(1,f)+k*w(2,f)-.25 OR i*w(1,f)*w(1,f)+j*w(1,f)+k*w(2,
f)+.25 THEN GO TO 750
590 NEXT f: LET i=(INT (i*10))/10: LET j=(INT (j*10))/10: LET k=(INT (k*10))/10
600 PRINT " Virfurile parabolilor sint pe o parabola a carei ecuatie este:
(" ;j;" ) x + (" ;j;" ) x + (" ;k;" ) = y"
610 LET h$="i*x*x+j*x+k"
620 PAUSE 0: GO TO 660
630 IF INKEY$="D" OR INKEY$="d" THEN STOP
640 IF INKEY$="n" OR INKEY$="N" THEN PAUSE 0: CLS : GO TO 5000
650 GO TO 630

```



```

660 RANDOMIZE USR 50112
670 IF swcirc=1 THEN GO TO 700
680 LET x=x1: PLOT INK 2; x+127,87+VAL f$
690 LET x=x2: PLOT INK 2; x+127,87+VAL f$
700 FOR x=-127/t TO 127/t STEP 1/t
710 LET y=VAL h$
720 IF y*t<-87 OR y*t>87 THEN GO TO 740
730 PLOT INK 2; x*t+127,y*t+87
740 NEXT x: PAUSE 0: GO TO 280
750 RANDOMIZE USR 50112
760 GO TO 280
785 REM INCEPUT.1dir: def UDG.Meniu
786 FOR f=50100 TO 50123: READ b: POKE f,b: NEXT f
789 DATA 17,0,167,33,0,64,1,0,27,237,176,201,17,0,64,33,0,167,1,0,27,237,176,20
1
799 CLS : FOR f=0 TO 55: read b: POKE 65368+f,b: NEXT f: RESTORE
800 DATA BIN 1110000,BIN 10010000,BIN 1000000,BIN 10000000,BIN 11111000,0,0,0,0,0
,0,0,0,BIN 11111000,BIN 10101000,BIN 10101000
803 DATA BIN 10000,BIN 110000,BIN 110000,BIN 1110000,BIN 11111000,BIN 11001100,BIN 1000011
0,BIN 11,BIN 11,BIN 11,BIN 11,BIN 10000110,BIN 11001100,BIN 11111000,BIN 1110000
,BIN 110000,BIN 10000
805 DATA BIN 1,BIN 10,BIN 100,BIN 11111111,BIN 10000,BIN 11111111,BIN 1000000,B
IN 10000000,0,BIN 11110,BIN 110000,BIN 1000000,BIN 1111110,BIN 1000000,BIN 11000
0,BIN 11110
806 DATA 0,0,BIN 1100110,BIN 10011001,BIN 10011001,BIN 10010110,BIN 1100000,0
5000 PAPER 6: INK 1: BORDER 4: CLS
5005 PRINT AT 9,11;"OPTIUNI"
5010 PRINT AT 5,1;"1 - Exemple"
5015 PRINT AT 7,1;"2 - Grafice"
5017 PRINT AT 9,1;"3 - Probleme"
5020 IF INKEY$="3" THEN GO TO 7000
5030 IF INKEY$="2" THEN GO TO 5500
5040 IF INKEY$="1" THEN GO TO 6000
5050 GO TO 5020
5500 GO TO 200
6000 CLS : PRINT AT 2,1;"1 - Graficele nu au puncte comune";AT 5,1;"2 -
Graficele au un punct comun";AT 7,1;"3 - Graficele au doua puncte comune
";AT 10,1;"4 - Graficele au virfurile pe o dreapta ";AT 13,1;"5 - Graficele
au virfurile pe o parabola"
6005 PRINT AT 16,1;"ENTER - Meniu"
6010 IF INKEY$="1" THEN GO TO 6500
6020 IF INKEY$="2" THEN GO TO 6600
6030 IF INKEY$="3" THEN GO TO 6700
6040 IF INKEY$="4" THEN GO TO 6800
6050 IF INKEY$="5" THEN GO TO 6900
6055 IF INKEY$=CHR$ 13 THEN GO TO 5000
6080 GO TO 6010
6500 LET m$="m": LET n$="2": LET p$="2+m": GO TO 6605
6600 LET m$="m*(m-1)": LET n$="m-2": LET p$="3"
6605 LET t=3
6606 LET f$="("+m$+"))*x*x+("+n$+"))*x+p$": CLS : PRINT AT 8,0; " A=";m$: PRINT "
B=";n$: PRINT " C=";p$: LET v=-0.2: LET w=.2: PRINT AT 12,3;"f (x)=";m$;"x +
";n$;"x";p$: LET sw=0: PAUSE 0: GO TO 10
6700 LET m$="m": LET n$="m+1": LET p$="-1-6*m": LET t=3: LET v=-.2: LET w=.2: PA
USE 0: GO TO 6605

```



```

6800 LET m$="m": LET n$="2*(m-1)": LET p$="m-1": GO TO 6605
6900 LET m$="(m+1)/2": LET n$="2*m": LET p$="(m-1)/(m+1)": LET t=10
6905 GO TO 6605
7000 CLS
7010 PRINT AT 5,1;"1 - Rezolvate"
7012 PRINT AT 7,1;"2 - Propune spre rezolvare"
7015 PRINT AT 9,1;"ENTER - Meniu"
7020 IF INKEY$="1" THEN GO TO 7100
7025 IF INKEY$="2" THEN GO TO 8000
7030 GO TO 7020
7100 PAPER S: INK 1: BORDER S: CLS
7110 FOR u=1 TO 3: CLS
7115 PRINT x$(u, TO): LET m$=v$(u, TO): LET n$=u$(u, TO): LET p$=v$(u, TO):
LET sw=1: LET f$="m$*x*x+n$*x+p$
7120 LET t=3: LET v=.3: LET w=.3: PAUSE 0: BEEP .1.29: GO TO 10
7200 CLS: PRINT r$(u, TO): PAUSE 0
7300 PRINT #0:AT 0,0:"Trecem la urmatoarea problema ? (D/N)"
7350 IF INKEY$="D" OR INKEY$="d" THEN NEXT u
7375 IF INKEY$="N" OR INKEY$="n" THEN GO TO 7000
7400 GO TO 7350
8000 PAPER S: INK 1: BORDER S: CLS
8010 FOR u=1 TO 3: CLS
8015 PRINT u$ - "y$(u, TO)
8020 PRINT #0:"Tastati orice ": PAUSE 0: BEEP .1.29
8030 NEXT u
8040 GO TO 5000

1 REM LOCURI GEOMETRICE
2 REM AUTORI:prof. DORIN MANI
3 REM prof. DOREL MINET
10 PAPER 0: BORDER 0: INK 7: CLS
15 LET $0=""
30 CLS
40 PRINT AT 2,3:"AUTORI: prof. MANI DORIN";AT 3,11:"prof. MINET DOREL";AT 5,10
1 Tema lectiei:";AT 7,6: "Determinarea unor locuri geometrice de baza"
50 PRINT AT 12,5: "Programul realizeaza:";AT 14,7:"-trasare locuri"
60 PAUSE 0: CLS: PRINT AT 1,14:"OPTIUNI";AT 4,7:"1 -loc geometric 1";AT 6,7:"
2 -loc geometric 2";AT 8,7: "3 -loc geometric 3";AT 10,7:"4 -loc geometric 4";AT
12,7: "5 -loc geometric 5";AT 14,7:"6 -loc geometric 6"
65 LET a$=INKEY$
70 PRINT #0:AT 0,6:"Alegeti optiunea:"
80 IF CODE a$<49 OR CODE a$>54 THEN GO TO 65
90 PRINT AT 2*VAL a$+2,7: FLASH 1;VAL a$
100 GO SUB VAL a$+1000
110 PAUSE 0: CLS: INPUT #0:"Doriti alta problema?";b$
130 IF CODE b$=100 OR CODE b$=68 THEN GO TO 60
150 STOP
1000 DIM a$(250): PAPER 1: INK 7: BORDER 3: CLS
1030 LET a$=" Fie !AB! un diametru fix al unui cerc C(0,r) iar M un punct var
iabil pe cerc. Se ia pe raza !OM! un punct P astfel incit !OP! se fie egala
cu distanta de la M la dreapta AB. Sa se afle locul geometric al punctului P."
1040 PRINT: PRINT: PRINT: PRINT: PRINT: PRINT: FOR i=1 TO LEN a$: PRINT a$
(i);: BEEP .01,10:NEXT i:PAUSE 0: CLS
1050 GO SUB 7000: LET xc=65: LET yc=85: LET r=60
1060 CIRCLE xc, yc, r: CIRCLE xc,yc,2: PLOT xc-r,yc: DRAW 2*r,0
1070 FOR a=PI/4 TO 2*PI STEP PI/4
1080 LET xa=xc+r*COS a: LET ya=yc+r*SIN a
1090 PLOT xa,ya: DRAW 0,yc-ya
1100 PLOT xc,yc: DRAW xa-xc, ya-yc
1110 LET xp=xc+ABS(ya-yc)*COS a: LET yp=yc+ABS(ya-yc)*SIN a
1120 INK 3: PLOT xc,yc: DRAW xp-xc,yp-yc: PLOT xc,yc: DRAW 0,yc-ya
1130 CIRCLE xp,yp,1

```

ANEXE



313



```

3230 RETURN
3240 FOR a=0 TO 2*PI STEP PI/4
3250 LET xm=xc+r*COS a: LET ym=yc+r*SIN a
3260 LET d=SQR (ABS (xm-xb)^2+ABS (ym-yb)^2)
3270 LET d1=SQR (ABS (xm-xa)^2+ABS (ym-ya)^2)
3280 LET d2=d+d1
3290 IF xm=xa THEN GO TO 3340
3300 LET b=ATN ((ym-yc)/(xm-xa))
3310 LET xp=xa+d2*COS b: LET yp=yc+d2*SIN b
3320 INK 3: PLOT xm,ym: DRAW xb-xm, yb-ym: INK 7: PLOT xa, ya: DRAW xm-xa, ym-ya:
INK 3: DRAW xp-xm, yp-ym: PAUSE 0
3330 GO SUB 3350: BEEP .05,-20: CIRCLE xp,yp,1: INK 7
3340 NEXT a
3350 INK 3: PLOT xm,ym: DRAW OVER 1: xb-xm, yb-ym: INK 7: PLOT xa, ya: DRAW OVER
1: xm-xa, ym-ya: INK 3: DRAW OVER 1: xp-xm, yp-ym
3360 RETURN
4000 PAPER 1: BORDER 3: INK 7: CLS
4010 DIM a$(150)
4020 LET a$=" Se considera un cerc, o coarda fixa AB si un punct M variabil pe cerc
Sa se determine locul geometric al ortocentrului triunghiului AMB.

4030 PRINT : PRINT : PRINT : PRINT : PRINT : PRINT : PRINT : FOR i=1 TO LEN a$:
PRINT a$(i);: BEEP .05,-30: NEXT i
4040 PAUSE 0: CLS
4050 GO SUB 7000: LET xc=120: LET yc=110: LET r=40
4060 LET xa=90: LET ya=yc-SQR (r^2-ABS (xc-xa)^2): LET xb=150: LET yb=ya: CIRCLE
xc,yc,r: PLOT xa, ya: DRAW xb-xa, yb-ya
4070 GO SUB 4170
4080 PAUSE 0
4090 FOR a=0 TO 2*PI STEP .1
4100 LET xm=xc+r*COS a: LET ym=yc+r*SIN a: CIRCLE xm,ym,1
4110 IF ym=yb THEN PRINT AT 0,0: "da": GO TO 4150
4120 LET a=(xb-xa)/(ya-yb)
4130 LET xh=xm: LET yh=ya+m*(xm-xa)
4140 BEEP .01,20: INK 3: PLOT xh,yh: INK 7
4150 NEXT a
4152 LET c$=" Locul geometric: un cerc simetric cu cercul dat fata de coarda data
4155 LET i=1: LET h=9: LET pas=9: LET deunde=5: LET pinaunde=12: LET unde=22
4157 PAUSE 0: GO SUB 1300
4160 RETURN
4170 FOR a=0 TO 5*PI/3 STEP 2*PI/9
4180 LET xm=xc+r*COS a: LET ym=yc+r*SIN a: LET a=(xb-xm)/(ym-yb): LET xh=xm: LET
yh=ya+m*(xm-xa)
4190 CIRCLE xm,ym,1: PLOT xa, ya: DRAW xm-xa, ym-ya: DRAW xb-xa, yb-ya: CIRCLE xh,yh,
1: BEEP .1,20: PAUSE 0: LET xha=xh: LET yha=yh: LET xma=xa: LET yma=ya: INK 3:
CIRCLE xma,yma,1: CIRCLE xha,yha,1
4200 GO SUB 4230: INK 7
4210 NEXT a
4220 RETURN
4230 PLOT xa, ya: DRAW OVER 1: xm-xa, ym-ya: DRAW OVER 1: xb-xa, yb-ya
4240 RETURN
5000 PAPER 1: BORDER 3: INK 7: CLS
5010 DIM a$(160)
5020 LET a$="Se considera un cerc, o coarda fixa AB si un punct M variabil pe
cerc. Sa se determine locul geometric al centrului de greutate al triunghiului AMB.
5030 PRINT : PRINT : PRINT : PRINT : PRINT : PRINT : PRINT : FOR i=1 TO LEN a$:
PRINT a$(i);: BEEP .05,-30: NEXT i
5040 PAUSE 0: CLS
5050 GO SUB 7000: LET xc=90: LET yc=100: LET r=60
5060 LET xa=50: LET ya=yc-SQR (r^2-ABS (xc-xa)^2): LET xb=130: LET yb=ya: CIRCLE
xc,yc,r: PLOT xa, ya: DRAW xb-xa, yb-ya

```



```

2350 LET z=yq: LET yq=yp: LET yp=z: LET t=xq: LET xq=xp: LET xp=t: GO SUB 2450:
IF xm=1 AND xm<=254 AND ym=1 AND ym<=174 THEN PLOT OVER 1; xm,ym
2360 IF d=r THEN DRAW xp-xm,yp-ym: PLOT xm,ym: DRAW xq-xm,yq-ym: PAUSE 0: GO SU
B 2430: PAUSE 0: GO TO 2380
2370 IF xm= AND xm<=254 AND ym=1 AND ym<=174 THEN DRAW xa-xm,ya,ym: PLOT xm,ym
: DRAW xb-xm,yb-ym: PAUSE 0: LET z=xp: LET xp=xa: LET xa=z: LET t=yp: LET yp=ya
: LET ya=t: LET u=xq: LET xq=xb: LET xb=u: LET v=yq: LET yq=yb: LET yb=v: GO SUB 2430:
LET xa=xp: LET ya=yp: LET xp=xq: LET yb=yq: GO TO 2390
2380 LET z=yp: LET yp=yq: LET yq=z: LET t=xp: LET xp=xq: LET xq=t: GO SUB 2560
2390 NEXT a
2400 RETURN
2410 PLOT xp,yp: DRAW OVER 1;xb-xp,yb-yp: PLOT xq,yq: DRAW OVER 1; xa-xq,ya-yq
2420 RETURN
2430 PLOT xm,ym: DRAW OVER 1; xp-xm,yp-ym: PLOT xm,ym: DRAW OVER 1;xq-xm,yq-ym
2440 RETURN
2450 IF xq=xa THEN GO TO 2570
2460 LET a1=(yq-ya)/(xq-xa)
2470 IF xp=xb THEN GO TO 2570
2480 LET a2=(yp-yb)/(xp-xb)
2490 IF a1=a2 THEN GO TO 2570
2500 LET xa=(ya-yb-a1*xa+a2*xb)/(a2-a1): LET ya=a1*xa+ya-a1*xa'
2510 IF xa=1 AND xa<=254 AND ya=1 AND ya<=174 THEN LET i=i+1: LET a(i)=x0: LE
T b(i)=y0: GO TO 2530
2520 GO TO 2570
2530 IF i/2=INT(i/2) THEN BEEP.01,20: INK 3: GO TO 2560
2540 INK 7
2550 BEEP.01,40
2560 PLOT xm,ym
2570 RETURN
2580 INK 5: PLOT xp,yp: DRAW OVER 1;xq-xp,yq-yp
2800 RETURN
3000 PAPER 1: BORDER 3: INK 7: CLS
3010 RESTORE 3010: FOR i=0 TO 7: READ x: POKE USR " "+i,x: NEXT i: DATA 24,32,64
,64,120,64,32,24
3020 DIM a$(150)
3030 LET a$=" !AB! fiind o coarda fixa, iar M un punct variabil al unui cerc, sa
se afle locul geometric al punctului P astfel incit !AP! si !MP!=!MB!.
3040 > PRINT :PRINT :PRINT :PRINT :PRINT :PRINT :FOR i=1 TO LEN a$:PRINT a$(i);:BEEP
. 05,-20:NEXT i
3050 PAUSE 0: CLS
3060 GO SUB 7000: LET r=30
3070 LET xc=100: LET yc=80
3080 LET xa=xc-r: LET ya=yc
3090 LET xb=xc+r/2: LET yb=yc+SQR(r^2-ABS (xb-xc)^2)
3100 CIRCLE xc,yc,r: PLOT xa,ya: DRAW xb-xa,yb-ya
3110 GO SUB 3240: GO SUB 3350
3120 FOR a=0 TO 3*PI STEP .1
3130 LET xa=xc+r*COS a: LET ya=yc+r*SIN a
3140 CIRCLE xa,ya,1
3150 LET d=SQR (ABS (xa-xb)^2*ABS (ya-yb)^2)
3160 LET d1=SQR (ABS (xa-xc)^2+ABS (ya-yc)^2)
3170 LET d2=d+d1
3180 LET b=ATN ((ya-yc)/(xa-xc))
3190 LET xp=xa+d2*COS b: LET yb=yc+d2*SIN b.
3200 BEEP .05,-20: PLOT xp,yp
3210 NEXT a
3220 PLOT xa,10: DRAW 0,150: PRINT AT 2,xa/8-2;"t"
3222 LET c$=" Locul geometric: doua portiuni de cercuri secante limitate de tan
genta in a la cerc care au coarda AB fixa, comuna.
3225 LET i=1: LET h=9: LET pas=9: LET deunde=5: LET pinaunde=17: LET unde=22
3226 PAUSE 0: GO SUB 1300

```



```

5070 GO SUB 5170
5080 PAUSE 0
5090 FOR a=0 TO 2*PI STEP .1
5100 LET xm=xc+r*COS a: LET ym=yc+r*SIN a: CIRCLE xm,ym,1
5110 IF ym=yb THEN PRINT AT 0,0: "da": GO TO 5150
5120 LET m=(xb-xm)/(ym-yb)
5130 LET xh=(xa+xb+xm)/3: LET yh=(ya+yb+ym)/3
5140 BEEP .01,20: INK 3: PLOT xh,yh: INK 7
5150 NEXT a
5151 LET c$=" Locul geometric: un cerc cu centrul la jumatatea distantei
dintre centrul cercului dat si coarda data si de raza 1/3 din raza cercului dat.
5152 LET i=1: LET h=10: LET pas=10: LET deunde=5: LET pinaunde=19: LET unde=20
5153 PAUSE 0: GO SUB 1300
5160 RETURN
5170 FOR a=0 TO 5*PI/3 STEP 2*PI/9
5180 LET xm=xc+r*COS a: LET ym=yc+r*SIN a: LET m=(xb-xm)/(ym-yb): LET xh=(xa+xb+
xm)/3: LET yh=(ya+yb+ym)/3
5190 CIRCLE xm,ym,1: PLOT xa,ya: DRAW xm-xa,ym-ya: DRAW xb-xm,yb-ym: CIRCLE xh,yh,
1: BEEP .01,20: PAUSE 0: LET xha=xh: LET yha=yh: LET xma=xm: LET yma=ym: INK 3
: CIRCLE xma,yma,1: CIRCLE xha,yha,1
5200 GO SUB 5230: INK 7
5210 NEXT a
5220 RETURN
5230 PLOT xa,ya: DRAW OVER 1: xm-xa,ym-ya: DRAW OVER 1: xb-xm,yb-ym
5240 RETURN
6000 PAPER 1: BORDER 5: INK 7: CLS
6010 DIM a$(200)
6020 LET a$="Fie A un punct fix, iar P un punct variabil al unui cerc. Sa se
afle locul geometric al punctului M de intersectie a bisectoarei unghiului POA
cu cercul circumscris triunghiului POA."
6030 PRINT : PRINT : PRINT : PRINT : PRINT : PRINT : FOR i=1 TO LEN a$:
PRINT a$(i): BEEP .05,-20: NEXT i
6040 PAUSE 0: CLS
6050 GO SUB 7000: LET xc=60: LET yc=80: LET r=40: LET xa=140: LET ya=yc: CIRCLE
xa,ya,1: PLOT xc,yc: DRAW xa-xc,ya-yc
6060 FOR a=.6 TO 1.9 STEP .2
6070 CIRCLE xc,yc,r
6080 LET xp=xc+r*COS a: LET yp=yc+r*SIN a: CIRCLE xp,yp,1
6090 PLOT xc,yc: DRAW xp-xc,yp-yc: DRAW xa-xp,ya-yp
6100 LET a=-1/TAN a: LET c=(xc+xp)/2: LET b=(yc+yp)/2: LET xr=(xc+xa)/2: LET yr=
b+a*(xr-c)
6110 LET r1=SQR (ABS (xr-xc)^2+ABS (yr-yc)^2)
6120 CIRCLE xr,yr,r1
6130 LET d1=(xa+r-xc)/2: LET xa=xc+d1: LET ya=yc+d1*TAN(a/2): PLOT xc,yc: DRAW
xa-xc,ya-yc: INK 3: CIRCLE xa,ya,1: BEEP .05,20: INK 7
6140 PAUSE 0
6150 PLOT xc,yc: DRAW INVERSE 1: xa-xc,ya-yc: CIRCLE OVER 1: xr,yr,r1: PLOT xa,ya
: DRAW INVERSE 1: xp-xa,yp-ya: DRAW INVERSE 1: xc-xp,yc-yp: CIRCLE xa,ya,1: PAUS
E 0: CIRCLE xa,ya,1
6160 NEXT a
6170 PAUSE 0
6180 FOR a=-PI/2+.1 TO PI/2+.3 STEP .1
6190 LET xp=xc+r*COS a: LET yp=yc+r*SIN a
6200 LET a=-1/TAN a: LET c=(xc+xp)/2: LET b=(yc+yp)/2: LET xr=(xc+xa)/2: LET yr=
b+a*(xr-c)
6210 LET r1=SQR (ABS (xr-xa)^2+ABS (yr-ya)^2)
6220 LET d1=(xa+r-xc)/2: LET xa=xc+d1: LET ya=yc+d1*TAN (a/2): CIRCLE xp,yp,1:
INK 3: CIRCLE xm,ym,1: BEEP .02,20: INK 7
6230 NEXT a
6235 LET c$=" Locul geometric: mediatoarea segmentului exterior cercului.
6240 LET i=1: LET h=9: LET pas=9: LET deunde=5: LET pinaunde=12: LET unde=20
6245 PAUSE 0: GO SUB 1300 \
6300 RETURN
7000 PLOT 0,0: DRAW 255,0: DRAW 0,175: DRAW -255,0: DRAW 0,-175
7010 RETURN

```



## Tema nr. 10

Rezolvarea acestei teme presupune mai multe etape:

1. Se citește funcția al cărei grafic se trasează, implementînd eventual în această etapă verificarea alfabetului folosit.

Alfabetul acceptat poate fi, de exemplu, format din:

- cifre: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9;
- operatori: +, -, \*, /, (ridicare la putere);
- funcții: SIN, COS, TAN, ASN, ACS, ATN, SQR, ABS, EXP, LN
- simboluri: X, a, b, c, d, e, f, g, h;
- delimitatori: (, ), . (punct zecimal).

Verificarea apartenenței caracterelor din șirul funcției la acest alfabet se poate face în etapa a doua, simultan cu analiza lexicală, sau în această etapă, în acest din urmă caz, fiind însă necesară eventual citirea șirului funcției, caracter cu caracter, implementîndu-se și cîteva funcții elementare de editare (poziționare cursor, ștergere, inserare).

Menționăm că, pentru simplificarea algoritmului, funcțiile acceptate de alfabet vor fi tastate folosind modul E (extins) — pe cursor E (se obține tastînd simultan CS (caps shift) + SS (simbol shift)).

II. Etapa aceasta se identifică cu etapa de analiză lexicală, care furnizează informația necesară etapei următoare de analiză gramaticală sau sintactică (verificarea unei gramatici), fiind asimilabilă cu o preprocesare a șirului funcției în vederea codificării lui în atomi lexicali.

Atomul lexical este (aici !) o entitate care poate fi privită ca un întreg de un anumit tip (vezi mai jos), cu care putem opera verificarea unei gramatici.

Pentru acest caz particular se vor identifica și codifica următoarele tipuri de atomi lexicali:

a) tip 0: **constante numerice** (numere întregi sau reale în format zecimal). Exemple: 0, PI, 4.753, 0.986, 123.0, 137.0603...

Nu se acceptă pentru simplificare) forme de genul: .35, 1.0739E+09...

b) tip 1: **simboluri (parametri de inserat)**

Se consideră că în expresia funcției se pot introduce parametri de inserat, în acest caz obținîndu-se de fapt o familie de funcții. Parametrii vor fi identificați automat de analizorul lexical (sînt maxim 8, dar numărul lor poate crește pînă la 25) și acesta va solicita, la fiecare nouă apariție a unui parametru, valoarea acestuia.

Pentru o dezvoltare ulterioară a programului, se va implementa trasarea graficului unei familii de funcții dependente de un număr de parametri, de inserat fiecare parametru putînd lua un set de valori.

Exemplu: a, b, c ..., x.

Nu se acceptă (tot pentru simplificare) identificatori de 2 sau mai multe caractere: a1, b72, ab1...

În această clasă am introdus și simbolul x, care are însă o semnificație bine determinată și anume aceea de variabilă pentru funcția data  $f=f(x)$ .

c) tip 2: **operatori**

Mulțimea operatorilor este formată din: +, -, \*, /, ^,

d) tip 3: **funcții standard elementare**

Mulțimea funcțiilor standard elementare este: SIN, COS, TAN, ASN, ACS, ATN, SQR, ABS, EXP, LN.

În expresia funcției se va considera că funcțiile standard elementare apar într-una din următoarele forme:

f (expresie):  $\text{SIN}(4 * x) = \sin(4x)$

f(x):  $\text{COS}x = \cos x$

f(-x):  $\text{SQR}-x = \sqrt{-x}$

realizîndu-se o implementare corespunzătoare a gramaticii (sintaxei) expresiei funcției.

e) tip 4: **delimitator stînga** = (

f) tip 5: **delimitator dreapta** = )



Vom considera eroare oricare din următoarele situații, ușor de identificat:  
 — mai mulți atomi lexicali de tip 4 decât cei de tip 5 sau invers (dezechilibrul balanței parantezelor)

— situația  $---(---(---))---$  — paranteze închise incorect

— situația  $---()---$  — absența unei expresii între paranteze.

De asemenea, pentru a facilita calculul și trasarea graficului se vor detecta, semnală și trata cazurile particulare:

$$f(x) = \text{const.} = k, k \in \mathbb{R}$$

$$f(x) = mx + n, m, n \in \mathbb{R}$$

$$f(x) = ax^2 + bx + c, a, b, c \in \mathbb{R}.$$

La ieșirea analizorului lexical se va furniza informația într-un tablou (vector) cu următoarea structură: A\$ (indice), unde:

< indice > este indicele atomului lexical, iar A\$ (indice) are structura

: atom lexical : — : tip : poziție :

unde < atom lexical > indică un șir care conține expresia atomului lexical (va'oaarea sa)

< tip > reprezintă tipul sau: 0—5

< poziție > reprezintă poziția sa în șirul funcției.

III. Această etapă, de analiză sintactică (gramaticală) preia datele furnizate de analizorul lexical (atomii lexicali) și, pe baza unei diagrame (graf) detectează eventualele erori de sintaxă din expresia funcției.

În acest caz particular, funcția poate fi verificată într-o singură trecere (parcurs) a șirului funcției, de fapt parcurgând șirul atomilor lexicali în ordinea în care aceștia apar în expresia funcției.

Această diagramă trebuie construită astfel:

a) După un atom lexical de tip 0 sau 1 (constantă numerică sau identificator) pot urma:

— un delimitator dreapta — tip 5

— un operator — tip 2

b) După un atom lexical de tip 2 (operator) pot urma:

— un identificator sau constantă numerică — tip 0

— un delimitator stînga — tip 4

— o funcție standard elementară — tip 3

c) După un atom lexical de tip 5 (delimitator dreapta) pot urma:

— un alt delimitator dreapta — tip 5

— un operator — tip 2

d) După un atom lexical de tip 4 (delimitator stînga) pot urma:

— un alt delimitator stînga — tip 4

— un identificator sau constantă numerică — tip 0

— o funcție standard elementară — tip 3

— semnul '—' (clasa operator — tip 2) într-un caz de excepție (discutat anterior)

e) După un atom lexical de tip 3 (funcție standard elementară) pot urma:

— un delimitator stînga (uzual) — tip 4

— un identificator sau constantă numerică — tip 0 sau 1, în cazul în care se acceptă situația de tipul ...SQRx... în loc de ...SQR(x)...

— o altă funcție standard — tip 3, în cazul ...SIN ABSx...

— semnul '—' (clasa operator — tip 2) în cazul de excepție discutat anterior.

Dacă din analiza făcută în această etapă nu au rezultat erori se trece la etapa următoare.

IV. Detectarea expresiilor care ar putea da erori la evaluarea numerică este necesară în această etapă, deoarece de exemplu, pentru funcția  $f(x) = 1/x$  și intervalul  $x \in [-5, 5]$ , în imediata vecinătate a lui  $x = 0$  evaluarea numerică va eșua (depășire superioară).

Pentru a evita situațiile de acest gen vom enumera cazurile care pot da nedeterminări:



- 1)  $\dots/E(x)\dots$ , cînd  $E(x)=0$
- 2)  $\dots SQR(E(x))\dots$ , cînd  $E(x) < 0$
- 3)  $\dots TAN(E(x))\dots$ , cînd  $COS(E(x))=0$
- 4)  $\dots LN(E(x))\dots$ , cînd  $E(x) \rightarrow 0$
- 5)  $\dots ASN(E(x))\dots$ , și
- $\dots ACS(E(x))\dots$ , cînd  $|E(x)| > 1$

Problema care trebuie rezolvată în această etapă este de detectare și extragere a expresiilor  $E(x)$  care apar într-una din situațiile de mai sus și încadrarea lor într-una din situațiile respective.

Exemple:

1) Dacă considerăm  $f(x)=LN SIN(1+1/x)$ , atunci în urma executării subprogramului corespunzător acestei etape vor rezulta următoarele expresii împreună cu tipul posibilelor erori și condițiile de eroare asociate:

$$1 \quad x \Rightarrow x = 0$$

$$4 \quad SIN(1+1/x) \Rightarrow SIN(1+1/x) \leq 0$$

2) Dacă considerăm  $f(x)=SQR(TAN(x+1)-1)$  atunci rezultă

$$3 \quad (x+1) \Rightarrow COS(x+1)=0$$

$$2 \quad (TAN(x+1)-1) \Rightarrow (TAN(x+1)-1) < 0.$$

Concret, rezultatul acestei etape va fi un număr de șiruri (de tipul celor subliniate) care conțin expresii logice. Aceste expresii vor fi evaluate în momentul trasării graficelor, pentru fiecare valoare a variabilei  $x$  în intervalul selectat  $[x_1, x_2]$ , și dacă toate sînt false, atunci se calculează punctul respectiv  $(x, f(x))$ . Dacă însă una din ele este adevărată, atunci, în funcție de tipul ei, în  $x=x_0$  corespunzător,  $f$  poate fi nedefinită, sau  $f$  are o asimptotă verticală.

V. După toate aceste etape pregătitoare, de analiză lexicală, sintactică, și de sinteză a expresiilor logice care pot genera erori, putem trece la calculul valorilor funcției, ținînd cont de rezultatul punctului IV și de un interval de valori  $x \in [x_1, x_2]$  selectat (în această etapă).

Pentru a evita unele probleme legate de scalarea automată, se vor calcula viteza de variație a funcției (derivata întâi) și viteza de variație a acesteia din urmă, adică accelerația (derivata a doua), folosind formulele cunoscute din metoda diferențelor finite cu pas constant.

Scalarea automată constă în alegerea unei valori  $|y| = y_{\max}$ , relativ la care se desenează graficul funcției (pe Oy). Alegerea se face din acele puncte așa-zis esențiale ale graficului funcției, pentru care  $|y_{n-1} - y_n| < \delta$ , unde  $\delta$  este valoarea maximă cu care poate varia viteza de variație a funcției, între două puncte consecutive.

Exemplu:  $f(x)=TANx$ ,  $x \in [0, 2\pi]$ .

Dreapta  $y=y_{\max}$  care rezultă din această evaluare înlătură prin „tăiere” partea nesemnificativă a graficului, evitînd o scalare automată greșită.

VI. După trasarea graficului funcției, se poate realiza o scalare pe Ox sau pe Oy prin selectarea grafică, interactivă, prin două drepte mobile, a coordonatelor de scalare, după care se reiau calculele.

VII. Dacă expresia funcției conține și parametri, atunci se va implementa o funcție de editare a acestora în scopul modificării valorilor și se trasează din nou funcția.

Programul SUPERGRAF elaborat\* în forma extinsă, a fost implementat cu ajutorul interpretorului BETA-BASIC, care asigură o viteză de calcul sporită și numeroase instrucțiuni și funcții suplimentare față de BASIC standard. Un exemplu de trasare a unui grafic cu SUPERGRAF este ilustrat în figura 16.1.

\* Realizator: Mihai Andrei, Mârșanu student la I.P.B., Facultatea de Automatică, anul II, 1988/1989.



Interfața 1, pe scurt IF1, înglobează de fapt pe o singură plachetă de circuit imprimat trei interfețe: cu discul flexibil (sau floppy), cu o linie serială standard CCITT V24 și o interfață mai puțin obișnuită care permite cuplarea mai multor HC-uri printr-o singură pereche de fire torsadate, conexiune denumită rețea.

Interfața de floppy permite cuplarea a până la două minidrive-uri (5,25" folii) cu 80 de piste, realizând o capacitate formată de 640 kiloocteti pe fiecare floppy disc, spațiu care poate fi utilizat pentru a memora până la 128 de fișiere distincte.

Rata medie de transfer a discului este de 25 până la 30 de ori mai mare decât a interfeței standard de casetă magnetică. Dacă mai adăugăm la aceasta și accesul aleator la informații (timpul maxim de acces la un sector de disc este de circa 1.7 secunde), este imposibil să nu remarcăm avantajele majore față de interfața de casetă.

Interfața serială rezolvă, în principal, problema cuplării unei imprimante la HC85, dar poate fi folosită și pentru a transfera date cu orice alt tip de calculator.

Interfața de rețea oferă o soluție pentru una din aplicațiile posibile ale calculatorului HC85: învățămîntul. Cuplate într-o rețea de până la 64 de sisteme, rețeaua HC-urilor dintr-o sală de laborator informatic poate ușura atât sarcina profesorului, cât și sarcina elevilor.

Viteza de transfer a informației prin rețea este de 80 kilobiti pe secundă. Transferurile de date se fac în blocuri cu lungime variabilă (maxim 255 octeti), însoțite de blocuri de control care specifică adresele sursă/destinație, număr bloc, etc. Protocolul este suficient de cuprinzător pentru a permite schimbul simultan de mesaje între oricâte noduri ale rețelei, folosind numai două fire torsadate pentru a lega nodurile între ele.

Din punct de vedere programe, IF1 se integrează în sistemul BASIC al calculatorului HC85, oferind fie noi instrucțiuni, fie extensii ale instrucțiunilor existente. Extinderea limbajului BASIC se face fără nici o modificare a plăcii de bază, interfața 1 interceptând prin hardware rutina de eroare din placa de bază.

Cele trei interfețe oferă pe lîngă facilități de încărcare, salvare programe și date, comenzi pentru manipularea de fișiere, ceea ce oferă o nouă dimensiune în stocarea și regăsirea datelor folosind programe scrise în BASIC.

#### UTILIZARE MINIDISC PENTRU PROGRAME

Salvare, verificare, încărcare și comasare programe

În manualul de utilizare HC85 am găsit instrucțiunea SAVE, care salvează programe pe casetă. Salvarea programelor pe floppy este la fel de simplă. Pentru exemplificare va fi folosit programul de mai jos, denumit Patrate. El tipărește numerele de la 1 la 10 împreună cu patratele lor.



```

10 REM Patrate
20 FOR n=1 TO 10
30 PRINT n,n*n
40 NEXT n

```

Pentru a salva acest program pe caseta, ati fi introdus:

```
SAVE "Patrate"
```

Pentru a-l salva pe floppyul din Minidrive-ul 1, introduceti:

```
SAVE "*"d";1;"Patrate"
```

Dupa citeva secunde in care marginea ecranului va clipi, programul va fi salvat.

(Numele programelor memorate pe floppy pot avea o lungime maxima de 11 caractere).

Asa cum probabil v-ati imaginat deja, puteti verifica corecta inregistrare a programului pe floppy introducind:

```
VERIFY "*"d";1;"Patrate"
```

Ecranul va afisa mesajul OK.

Puteti incarca acum programul Patrate introducind:

```
NEW
```

urmat de:

```
LOAD "*"d";1;"Patrate"
```

In continuare, pentru a face ca programul sa se lanseze automat, incercati sa introduceti:

```
SAVE "*"d";1;"Patrate2" LINE 10
```

apoi:

```
NEW
```

si apoi:

```
LOAD "*"d";1;"Patrate2"
```

Minidrive-ul poate fi folosit si pentru a comasa programele. Introduceti:

```
NEW
```

urmat de:

```

100 REM alte Patrate
110 FOR n=11 TO 20
120 PRINT n,n*n
130 NEXT n

```



si acum introduceti:

```
MERGE "d";1;"Patrate"
```

si programul Patrate va fi adaugat la listing.

Pe scurt, asa cum v-ati dat deja seama, sintaxa folosita pentru obisnuita interfata de caseta

se aplica si la Minidisc.

### Stergerea programelor

Se presupunem ca ati terminat de lucrat cu programul Patrate.

Pentru a-l sterge, introduceti:

```
ERASE "d";1;"Patrate"
```

(Ca inainte, "d";1 indica ce Minidrive folositi).

In timpul executiei instructiunii ERASE, marginea ecranului va clipi.

### Formarea discurilor

Inainte de prima utilizare a unui floppy, inserati-l intr-un Minidrive (de exemplu Minidrive-ul 1) si introduceti:

```
FORMAT "d";1
```

"d";1 identifica Minidrive-ul pe care il folositi (in acest caz Minidrive-ul 1).

Formarea unui disc dureaza aproximativ treizeci de secunde. In timpul acesta, marginea ecranului se va schimba la inceput, si va reveni putin inainte de afisarea mesajului OK. Procesul de formare consta din initializarea fiecărei piste de pe floppy, prin scrierea cimpurilor de identificare si date corespunzatoare fiecarui sector. Dupa formarea unei piste, fiecare sector in parte este citit, verificand suma de control. Mesajul OK apare numai daca toate sectoarele au putut fi citite corect (nu se accepta discuri cu sectoare eronate).

Formarea unui floppy nu trebuie repetata niciodata, si pentru ca prin formarea unui disc se pierde orice a fost inregistrat pe el. Apasati acum:

```
CAT 1
```

1 identifica numarul minidrive-ului pe care il folositi

Dupa cateva secunde, in care timp marginea ecranului va clipi, va apare mesajul de eroare:

```
File not found
```

care semnifica faptul ca floppyul nu contine nici un program.

Capacitatea unui disc este de 636 kiloocteti, plus 4 kiloocteti pentru zona de catalog, gestionata de sistem.

**PROLOG — DIALOG — EPILOG**



### Instalarea facilitatii de auto-run

Putin mai inainte ati folosit facilitatea de auto-run pentru discul de demonstratie. Daca aveti un program pe care il folositi adesea, va puteti stabili propria facilitate de auto-run, astfel incit sa nu mai introduceti instructiunile LOAD si RUN. Acestea sint regulile de urmat:

- programul trebuie sa aiba numele run;
- floppyul trebuie folosit in Minidrive-ul 1;
- facilitatea trebuie folosita fie imediat dupa punerea sub tensiune, sau imediat dupa comanda NEW.

Astfel, introduceti programul respectiv, urmat de comanda:

SAVE "\*"d";1;"run" LINE numar

+--- introduceti aici numarul  
liniei de start

+----- numele run trebuie introdus  
litera cu litera. Nu apasati  
tasta RUN

Acum introduceti:

NEW

urmat de:

RUN

+--- Tasta RUN, si nu numele programului.

### DATE, CANALE SI CAI

Precum stiti, un program este un set de operatii care se executa atunci cind apasati RUN. Datele, pe de alta parte, sint orice colectie de litere, numere sau simboluri cu care poate lucra un program. Exemple sint numerele de la 1 la 10 si patratele lor.

Datele pot fi trimise, sau receptionate, catre/de la diferite parti ale unui sistem de calcul. Aceste parti sint denumite "canale". Canalele catre care se pot trimite date sint:

- ecranul televizorului
- un fisier pe floppy
- un alt calculator HC85, daca amindoua calculatoarele sint cuplate printr-o retea.
- interfata RS232 si de acolo, de exemplu, la un modem sau o imprimanta.

Canalele de la care se pot primi date sint:

**HC-85 EXTINS, HC-88 (DISC, REȚEA)**



- claviatura
- un fisier pe floppy
- un alt calculator HC85, dacă amîndoua calculatoarele sînt cuplate printr-o retea.
- interfața RS232, adică un modem sau un terminal.

Nodurile de comunicare dintre programul BASIC și canale sînt denumite cai. În sistemul HC85, numărul acestor cai este fixat la 16. Ele sînt numerotate de la 0 la 15, iar numerele de cale sînt întotdeauna precedate de semnul #.

Patru dintre aceste cai sînt deja cuplate la următoarele canale:

- calea #0 trimite date către partea de jos a ecranului TV și
- calea #1 primește date de la claviatura;
- calea #2 trimite date către partea de sus a ecranului TV, dar nu poate primi date;
- calea #3 trimite date către imprimanta, dar nu poate primi date.

Orice instrucțiune care execută un transfer de intrare/ieșire folosește una din aceste cai în mod implicit. De exemplu, instrucțiunea PRINT folosește calea #2, iar instrucțiunea LPRINT folosește calea #3. Astfel, dacă introduceți:

PRINT "Acesta este un calculator HC85"

folosiți de fapt o prescurtare a instrucțiunii:

PRINT #2;"Acesta este un calculator HC85"

Verificați prin introducerea celor două forme.

Puteti, totuși, să faceți fiecare instrucțiune să folosească o altă cale prin introducerea semnului # urmat de un număr de cale. Încercați să introduceți:

LPRINT #2;"Acesta este un calculator HC85"

în loc să fie trimis la imprimantă, acest mesaj apare pe ecranul TV.

Dar în loc să folosiți caile prestabilite, puteți crea unele proprii. Căile #4 pînă la #15 sînt rezervate pentru acest scop; și există diferite 'specificatoare de canale' care indică perifericul dorit. Cîteva exemple sînt:

"K" pentru claviatura

"S" pentru ecran

"P" pentru imprimantă



(altele vor fi introduse mai tirziu).

Remarcati faptul ca K, S si P sint toate canale prestabilite. Ele solicita utilizarea virgulelor (,) drept separatori in instructiunile OPEN #. Dar cu alte canale puteti folosi fie virgule fie punct-virgula (;).

Pentru a creea o cale proprie folositi instructiunea OPEN #. De exemplu introduceti:

```
10 OPEN #4,"S"
```

Astfel deschideti calea #4 si o cuplati la canalul "S". Acum introduceti:

```
20 PRINT #4;"Acesta este un calculator HC85"
```

si din nou linia va apare pe ecran.

(Nu se recomanda deschiderea cailor 0, 1 sau 2, pentru ca rezultatele acestor operatii pot fi imprevizibile).

### FISIERE DE DATE PE DISC

#### Deschiderea unui fisier de date

Memorarea informatiilor pe floppy se face in fisiere. Fiecare fisier primeste la creare un nume, pentru a putea fi regasit mai tirziu. Instructiunea care deschide si denumeste un fisier de date are intotdeauna aceeaasi forma. De exemplu testati instructiunea:

```
OPEN #4;"d";1;"Numere"
```

"Numere" este numele fisierului. Acesta poate fi orice sir de caractere de lungime maxima 11

"d";1 identifica Minidiscul pe care il folositi

numarul de cale poate fi orice numar intre 0 si 15

Aceasta instructiune face doua lucruri distincte:

stabileste un canal cuplat cu fisierul: "d";1;"Numere"

ataseaza acest nou canal la calea #4.

Operatia va dura cateva secunde, in care timp calculatorul va cauta pe floppy un fisier cu numele "Numere". Pentru ca nu exista fisierul "Numere", deschide canalul pentru scriere. (Daca ar fi gasit un fisier cu numele "Numere", l-ar fi deschis pentru citire.)

#### Introducerea datelor

Odata ce ati deschis un fisier, puteti introduce date. Sa presupunem ca vreti sa memorati numerele de la 1 la 10 impreuna cu patratele lor. Introduceti si rulati programul urmator:



```

10 FOR n=1 TO 10
20 PRINT #4;n*n*n
30 NEXT n

```

S-ar putea sa credeti ca toate numerele au fost deja memorate pe floppy. Dar de fapt calculatorul nu transfera in mod automat datele pe floppy decat dupa ce s-a acumulat o anumita cantitate de informatii, pe care o transfera dintr-odata. Acest procedeu se numeste 'blocarea' datelor. Un bloc de date pe floppy are lungimea de 256 de octeti (sau caractere).

Pentru a memora pe floppy datele introduse trebuie sa inchideti fisierul. Pina nu faceti acest lucru, nu veti putea sa cititi din fisier.

#### Inchiderea unui fisier

Inchiderea unui fisier asigura memorarea definitiva a datelor pe floppy. Inchide de asemenea canalul (in cazul nostru "d";1;"Numere") si detaseaza calea (in cazul nostru #4) de la orice canal. Pentru a inchide un fisier trebuie doar sa inchideti calea asociata:

```
CLOSE #4
```

Marginea ecranului va clipi pentru a arata ca se inregistreaza ceva pe floppy.

(Remarcati faptul ca, la fel ca la instructiunea OPEN, instructiunea CLOSE este urmata in mod automat de #).

Caile #0, #1, #2, #3 ramin intotdeauna atasate unui canal, chiar daca se executa o instructiune CLOSE specifica. Daca incercati sa inchideti una din aceste cai, caile #0 si #1 se vor atasa automat la canalul K; calea #2 la canalul S; iar calea #3 la canalul P.

#### Citirea datelor dintr-un fisier

Pentru a citi datele din fisierul "Numere" rulati urmatorul program:

```

10 OPEN #4;"d";1;"Numere"
20 FOR b=1 TO 10
30 INPUT #4;m;n
40 PRINT "Patratul lui ";m;" este ";n
50 NEXT b
60 CLOSE #4
RUN

```

Pentru ca fisierul "Numere" exista deja pe floppy, canalul "d";1;"Numere" este deschis pentru intrare, si orice incercare de a scrie date ar fi generat o eroare.

Se poate deasemenea folosi functia INKEY\$ pentru a citi date dintr-un fisier (intoarce intotdeauna urmatorul caracter din fisier). Incercati programul urmator:



```

10 OPEN #11;"d";1;"listing"
20 LIST #11
30 CLOSE #11
40 OPEN #12;"d";1;"listing"
50 PRINT INKEY#12;
60 GO TO 50

```

Acest program se va termina cu un mesaj de sfirsit de fisier, adica End of file.

#### Observatii asupra lui PRINT si INPUT

Pentru ca instructiunile PRINT si INPUT au fost concepute in principal pentru utilizarea cu ecranul si claviatura, trebuie sa iiti atenti la folosirea lor cu fisiere.

separatori:

Instructiunea PRINT are trei forme de separatori:

- semnul ; (punct-virgula) nu tipareste nimic,
- semnul , (virgula) va aduce la inceputul urmatoarei jumatați de linie,
- semnul ' (apostrof) sare la linie noua (codul CR).

Instructiunea INPUT asteapta intotdeauna sa introduceti CR dupa un numar sau un sir. Astfel, de fiecare data cind tipariti intr-un fisier din care vreti sa cititi mai tirziu cu INPUT, trebuie fie sa:

- tipariti fiecare element separat, adica

```

10 PRINT #4;2
20 PRINT #4;3

```

sau

- separati elementele cu apostrof, adica

```
10 PRINT #4;2'3
```

Deasemenea, in instructiunile INPUT, trebuie sa folositi cu atentie separatorii. Aşa cum stiti, INPUT poate tipari in partea de jos a ecranului orice se poate pune intr-o instructiune PRINT. Dar daca cititi cu INPUT dintr-un fisier, fisierul se deschide numai pentru citire. Aşa incit, daca includeti orice s-ar fi tiparit la utilizarea ecranului, veti obtine mesajul de eroare Writing to a 'read' file (Scriere intr-un fisier de citire). Aceasta inseamna ca elementele dintr-o instructiune INPUT trebuie separate numai prin punct-virgula, adica

```
10 INPUT #4;a;b
```

Atentie de asemenea la citirea cu INPUT a sirurilor de caractere care contin " (ghilimele), pentru ca INPUT va interpreta ghilimelele drept sfirsit de sir. Metoda de a evita acest lucru este de a inlocui, de exemplu:

```
10 INPUT #4;a$ cu 10 INPUT #4;1 LINE a$
```



### Schimbarea cailor

Instrucțiunile PRINT pot conține informații pentru mai multe cai la un moment dat. Programul următor va tipări "unu" pe ecran; "doi" într-un fișier pe floppy denumit "cifre"; "trei" către stația 1 pe rețea (vezi capitolul următor); și "patru" în următoarea linie din ecran.

```
10 OPEN #4;"d";1;"cifre"  
20 OPEN #5;"n";1  
30 PRINT "unu";#4;"doi";#5;"trei";#2;"patru"  
40 CLOSE #4  
50 CLOSE #5
```

### 'schimbarea culorilor'

După ce ați folosit un canal diferit de ecran, se poate ca instrucțiunile PAPER și INK să nu aibă nici un efect. Pentru a evita acest lucru, introduceți:

PRINT;

înainte de a schimba PAPER sau INK.

### Afișarea catalogului de fișiere

Pe măsură ce se înregistrează fișiere pe floppy, acestea sunt introduse în mod automat în catalog. Astfel, pentru a afla ce fișiere sunt înregistrate pe floppy, este suficient să inserați floppyul într-un Minidrive și să introduceți instrucțiunea CATALOG. De exemplu, introduceți:

CAT 1

Ecranul televizorului va afișa:

- numele fișierelor
- spațiul disponibil rămas pe floppy (în kiloocteti)

Puteti să transferați ieșirea unui CAT către o cale introducând:

```
CAT #numar;numar  
      |      |  
      |      +----- numar Minidrive  
      |  
      +----- numar cale
```

Aceasta vă permite să trimiteți catalogul către a imprimantă, sau către un fișier, astfel încât să poată fi folosit de un program.

### Protejarea unui fișier

Dacă doriți ca un nume să nu apară în catalog, îl puteți proteja dându-i un nume care are în poziția 10 codul caracterului dorit plus 128. Introduceți acest program:



```

10 OPEN #4,"d";1;"Rezultate"+ CHR$(128+ CODE " ")
20 FOR n=1 TO 15
30 PRINT #4;n'n*n
40 NEXT n
50 CLOSE #4

```

Acum introduceti: CAT 1

Numele fisierului nu va apare. Astfel ca, de fiecare data cind generati un fisier protejat, amintiti-va sa-i notati numele undeva, pentru cazul in care ii uitati numele!

## PRIMELE OPERATII CU MINIDISCU

### Auto-run

Dupa ce ati terminat de instalat Interfata 1 si Minidrive-ul sinteti curios sa aflati ce programe va asteapta pe discul de demonstratie. Pentru asta, inserati floppy discul in Minidrive (sau daca aveti doua Minidrive-uri, in Minidrive-ul 1), si introduceti:

NEW      urmat de:    RUN (si CR)

Aceste comenzi vor declansa incarcarea automata si rularea primului program de pe floppy. Dupa ce ati terminat de privit acest program, cititi mai departe.

### Catalogul

Pentru a afla ce alte programe se gasesc pe floppyul de demonstratie, introduceti instructiunea CATalog:

```

CAT 1
+--- 1 identifica numarul Minidrive-ului pe care
      il folositi

```

In aproximativ 3 secunde pe ecranul televizorului se va afisa:

- un catalog al tuturor numelor fisierelor memorate pe floppy
- spatiul ramas disponibil pe floppy (in kiloocteti)

### Incercarea programelor

Urmatorul lucru de facut este incarcarea programului pe care vreti sa-l executati in continuare. Pentru asta alegeti mai intii un program, apoi introduceti:

```

LOAD *"d";1;"nume"
+--- aici introduceti numele programului
      pe care l-ati ales
+----- "d";1 identifica ce Minidrive
          folositi
+----- steluta comunica calculatorului ca
          folositi un Minidrive, si nu
          interfata obisnuita de caseta

```



Dupa o scurta pauza, ecranul va afisa mesajul OK (dar fara numele programului). Puteti acum lansa programul in executie (cu RUN).

### Extinderea unui fisier

Se presupunem ca vreti sa extindeti fisierul "Numere" pentru a include patratele numerelor de la 1 la 20 in loc de numai 1 la 10. Un fisier nu poate fi redeschis pentru scriere, astfel ca trebuie sa:

- creati o noua versiune cu alt nume;
- transferati vechiul fisier in noua versiune;
- adaugati noile date
- inchideti vechiul fisier.

Iata cum se poate face aceasta.

Mai intii rulati acest program:

```
10 OPEN #4;"d";1;"Numere": REM pentru citire
20 OPEN #5;"d";1;"Numere 1": REM pentru scriere
30 FOR f=1 TO 10
40 INPUT #4;m;n
50 PRINT #5;m'n
60 NEXT f
70 FOR n=11 TO 20
80 PRINT #5;n'n*n
90 NEXT n
100 CLOSE #4; CLOSE #5
```

Pentru a verifica existenta a doua fisiere, "Numere" si "Numere 1", introduceti:

CAT 1

Apoi, ca sa stergeti vechiul fisier, introduceti:

ERASE "d";1;"Numere"

Pentru a verifica stergerea, introduceti:

CAT 1

Numele fisierului "Numere" a disparut din catalog, iar noul fisier, "Numere 1" contine acum numerele de la 1 la 20:

### RETEAUA LOCALA



### Configurarea unei rețele

Rețeaua locală permite utilizatorului și prietenilor lui să schimbe între ei programe și date. Aceasta înseamnă că numai unul dintre voi trebuie să introducă un program. O rețea este foarte utilă și dacă numai unul dintre voi are un Minidisc.

Folosind cablul furnizat odată cu interfata, puteți lega de la două până la saizecisipatru de calculatoare HC85.

Configurarea rețelei nu trebuie să fie în nici un caz o buclă închisă: calculatoarele de la capetele rețelei nu trebuie să fie conectate între ele. Fiecare capăt de rețea trebuie să aibă un conector neocupat.

=====

NU PORNIȚI ȘI NU OPRITI NICIODATĂ UN HC85 CARE ESTE CUPLAT LA REȚEA ÎN TIMP CE ARE LOC UN TRANSFER DE DATE PE REȚEA. Totuși puteți avea un HC85 oprit pe rețea; puteți de asemenea să porniți sau să opriți HC85-uri care sînt pe rețea, cu condiția ca să nu se facă transferuri pe rețea în acel moment.

=====

După ce ați stabilit o rețea, fiecare calculator (sau stație) trebuie să primească un număr de identificare diferit. Mai întâi stabiliți împreună cu prietenii dumneavoastră, care va fi numărul fiecărei stații, după care fiecare dintre dumneavoastră trebuie să introducă:

```
FORMAT "n";numar
```

```
      +----- introduceti numarul de statie  
                        pe care l-ati ales.
```

Dacă rețeaua este formată numai din două calculatoare, amindouă pot folosi același număr de stație. Și pentru că, amindouă calculatoarele devin stația 1 în mod automat la punerea sub tensiune, utilizarea instrucțiunii FORMAT nu mai este necesară.

### Programele și rețeaua

Să presupunem că ați cuplat două calculatoare într-o rețea, cu numerele de stație 1 respectiv 2.

Să presupunem că vreți să trimiteți către stația 2 următorul program:

```
10 REM patrate  
20 FOR n=1 TO 10  
30 PRINT n,SQR n  
40 NEXT n
```

Introduceți programul urmat de:

```
SAVE "*"n";2
```

(Remarcați că rețeaua nu folosește nume pentru programe.)



Intre timp la statia 2 trebuie introdus:

```
FORMAT "n";2
```

urmat de:

```
LOAD "*"n";1
```

Statia 2 va avea acum o copie a programului. Remarcati cum marginea ecranului ramane neagra in timp ce calculatorul asteapta sa salveze sau sa incarce programul prin retea. Statia 1 nu va trimite pina cind statia 2 nu este gata, iar statia 2 va astepta pina cind se emite ceva. Incercati sa introduceti linia cu SAVE inainte ca la statia 2 sa introduceti LOAD si vice versa.

Pentru a verifica transmitia corecta a programului, la statia 2 trebuie introdus:

```
VERIFY "*"n";1
```

in timp ce la statia 1 se repeta transmitia programului introducind:

```
SAVE "*"n";2
```

SAVE este de fapt singura instructiune care transmite programe in retea. Instructiunile LOAD, VERIFY si MERGE sint toate metode de a receptiona programele.

Jocul de retea din Apendixul 1 este un bun exemplu pentru utilizarea programelor in retea.

### Fisiere de date in retea

Sa presupunem ca doriti sa transmiteti acum date catre statia 2. Instructiunea OPEN #4;"n";2 deschide un canal catre statia 2 pe retea si ataseaza calea #4 la el, astfel incit daca scrieti prin calea #4, mesajul va fi pus pe retea impreuna cu o notita care indica sursa mesajului.

Daca ati fi introdus INPUT #4;m\$ calculatorul dumneavoastra ar fi asteptat informatii adresate statiei 1 de la statia 2.

Acum introduceti acest program:

```
10 OPEN #4;"n";2: REM pentru iesire
20 INPUT a$: PRINT #4;a$
30 CLOSE #4
40 OPEN #4;"n";2: REM pentru intrare
50 INPUT #4;b$: PRINT b$
60 CLOSE #4
70 GO TO 10
```

Apoi introduceti:

```
SAVE "*"n";2
```

Acum introduceti la statia 2:

```
FORMAT "n";2
LOAD "*"n";1
```



Introduceti la statia 1 RUN, iar la statia 2 editati liniile 10 si 40 pentru a se referi la statia 1 si nu la statia 2. Apoi se introduce la statia 2:

GO TO 40

Sinteti acum gata sa incepeti o conversatie. Dar inainte de a face asta ar trebui sa stiti citeva lucruri.

- Tot ce tipariti prin calea #4 este blocat: adica nu este imediat pus pe retea, ci se asteapta pina cind se acumuleaza o anumita cantitate de date. Asa ca este necesara inchiderea canalului prin CLOSE, imediat ce ati terminat de tiparit. Astfel se transmite zona tampon chiar daca nu este plina. (Zona tampon are lungimea 255 de octeti sau caractere.)

Tot ce tipariti este marcat ca sosind in mod specific de la statia la care lucrati, astfel incit daca statia 2 este in asteptare pentru un mesaj de la alta statie, mesajul dumneavoastra va fi ignorat. Daca mesajul emis este ignorat, ecranul nu va afisa mesajul OK, si marginea ecranului va ramine neagra pina cind mesajul este emis si se primeste confirmare pozitiva de la statia 2.

- In timp ce instructiunea INPUT poate fi folosita pentru a astepta ca ceva sa fie transmis, functia INKEY\$ poate fi folosita pentru a citi retea. Se va intoarce cu primul octet din orice a fost transmis, sau din orice astepta sa fie transmis. Altfel se intoarce cu sirul vid. Aceasta se numeste interogare (polling).

Programul de mai jos va tipari orice este transmis catre statia 1:

```
10 OPEN #8;"n";1
20 PRINT INKEY$#8;
30 GO TO 20
```

### Emisie generala

Exista un numar de statie special, al carui specificator este "n";0. Atunci cind se asteapta date de la statia 0, veti receptiona orice mesaj care este emis catre statia 0. Iar atunci cind scrieti, mesajul emis catre statia 0 va fi receptionat de oricine citeste date de la un canal cu specificatorul "n";0.

Aceasta ar fi foarte util, de exemplu, intro scoala daca fiecare elev ar avea un calculator, dar numai profesorul ar avea un minidrive.

Sa presupunem ca profesorul doreste sa emita un program. Mai intii toti elevii ar trebui sa introduca: LOAD \*"n";0

Aceasta comanda va face ca toate calculatoarele elevilor sa intre in asteptare pentru receptia programului. Profesorul ar trebui sa salveze programul in retea introducind: SAVE \*"n";0

Emisiile generale, spre deosebire de mesajele private, incep imediat fara sa astepte ca alte calculatoare sa fie gata

De asemenea, la emisia generala, calculatorul nu va poate informa daca mesajul emis a fost receptionat de catre cineva.

Functia INKEY\$ nu poate fi folosita pentru a interoga un canal de receptie generala. La fel ca INPUT, ea va astepta pur si simplu ca ceva sa fie emis.



## UTILIZAREA INTERFETEI SERIALE

### Conectarea perifericelor la interfata seriala

Precum stiti, setul de caractere al lui HC85 contine atit simboluri simple (litere, cifre, etc.) cit si cuvinte cheie (instructiuni, nume de functii, etc.). Toate aceste caractere pot fi emise si receptionate prin interfata seriala catre/de la orice dispozitiv compatibil; de exemplu o imprimanta, un modem sau o alta interfata seriala conectata la un tip diferit de calculator.

Pentru a conecta oricare din aceste periferice la interfata seriala, trebuie sa folositi un cablu cu un conector cu 9 pini la capatul dinspre interfata 1 si un conector corespunzator dispozitivului la care va cuplati la celalalt capat. (Pentru detalii de interconectare vezi Appendixul 4.)

Apoi, inainte de a folosi interfata seriala, va trebui sa stabiliti modul de lucru al perifericului:

- modul 'auto line feed' trebuie dezactivat. (HC85 va emite secventa 'retur car' (CR) si 'avans rind' (LF) pe un canal "t", dar numai 'retur car' (CR) pe un canal "b". Aceste canale "t" si "b" sint explicate mai jos.)
- paritatea trebuie dezactivata.
- numarul de biti trebuie stabilit la 8 (opt).
- numarul de biti de stop trebuie stabilit la 1 (unu).
- viteza de emisie/receptie (adica numarul de biti pe secunda). HC85 poate comunica la oricare din vitezele standard, adica:  
50, 110, 300, 600, 1200, 2400, 4800, 9600, 19200.

Este bine sa folositi cea mai mare viteza pe care o permite perifericul la care va cuplati. (Veti vedea mai jos cum puteti face ca HC85 sa foloseasca aceasi viteza.)

In astfel de momente, un manual de instalare pentru perifericul la care va cuplati este foarte util.

### Canalele t si b

Interfata seriala are doua tipuri de canale: canalul t si canalul b.

#### Canalul t.

Canalul t (de la text) este folosit de obicei pentru a trimite listinguri. Canalul t are urmatorul efect asupra setului de caractere:

#### cod caracter

0-31 (caracterele de control) nu sint emise, cu exceptia lui 13 (retur car) care este trimis ca 13 urmat de 10 (retur car si avans rind).

32-126 (caracterele tiparibile) sint trimise ca atare.



127-164 (caracterele grafice) nu sînt emise. Ele sînt înlocuite de caracterul ? (codul 63).

Pentru INPUT și INKEY\$ canalul t transferă numai caractere pe 7 biți, așa încît forțează la 0 bitul 7.

Pentru a folosi canalul t, trebuie mai întîi să stabiliți viteza de lucru pe sîrme. Așa ca introduceți:

```
10 FORMAT "t";viteza
```

```
      |  
      +---- introduceți aici viteza de  
            lucru pe care ați stabilit-o  
            și la periferic.
```

Acum, pentru a deschide o cale către canalul t, introduceți:

```
20 OPEN #3,"t"  
30 LLIST
```

Marginea ecranului va clipi și listingul va fi trimis către periferic. (Remarcați că LLIST este o prescurtare de la [IST #3.]) Introduceți acum:

```
LPRINT "Acesta este un mesaj."
```

Si acest mesaj va fi trimis către dispozitiv.

Dacă HC85 este cuplat cu un terminal sau un calculator care poate trimite caractere, atunci puteți citi date de la terminal sau calculator. Introduceți:

```
10 FORMAT "t";viteza  
20 OPEN #4,"t"  
30 PRINT INKEY$#4;  
40 GO TO 30
```

Acum, orice caracter primit de la terminal sau calculator va fi afișat pe ecran.

Canalul b

Canalul b (de la binar) trimite toți cei 8 biți ai codurilor folosite de HC85, și va permite să trimiteți coduri de control către imprimante etc.

Si la INPUT și INKEY\$ canalul b întoarce caractere pe 8 biți.

SAVE și LOAD funcționează numai cu canalul b.

Dacă ați conectat două HC-uri prin interfața serială sau doriți să vă memorati programele pe un alt tip de calculator care are deasemenea o interfața serială, veți dori să salvați și să încarcați programe prin interfața serială. Pentru aceasta introduceți:

```
FORMAT "b",viteza
```

```
      |  
      +---- introduceți aici viteza pe care ați  
            stabilit-o la periferic
```



Acum puteti incerca, de exemplu:

```
10 REM cifre,  
20 FOR n=1 TO 10  
30 PRINT n,n*RND  
40 NEXT n
```

urmat de:

```
SAVE "*"b"
```

la celalalt capat al legaturii cineva trebuie sa introduca:

```
LOAD "*"b"
```

Extensiile uzuale sint deasemenea posibile:

```
SAVE "*"b";SCREEN$
```

si:

```
SAVE "*"b";LINE numar
```

### Cum se trimite coduri de control

Multe imprimante primesc secvente de control pentru operatii de genul tiparire cu latime dubla. Pentru a trimite caracterele de control trebuie sa folositi canalul b. Atentie insa, prin canalul b returul de car (CR) nu este urmat automat de avans rind (LF). De aceea este preferabil sa aveti doua canale deschise, unul b si altul t: veti folosi canalul b pentru a trimite secventele de control, si canalul t pentru a trimite textele.

De exemplu sa presupunem ca codul de control pentru tiparire cu latime dubla este 14. Introduceti:

```
10 OPEN #4;"b"  
20 PRINT #4;"Latime normala ";  
30 PRINT #4;CHR$ 14;"Latime dubla"  
40 CLOSE #4
```

(Daca exemplul nu functioneaza, cautati in manualul imprimantei codul pentru latime dubla.)

Incarcati si exemplul de mai jos:

```
10 OPEN #5;"b"  
20 OPEN #6;"t"  
30 PRINT #5; CHR$ 14;  
40 LIST #6  
50 CLOSE #5; CLOSE #6
```

Acest exemplu ar trebui sa produca un listing pe latime dubla.



## INSTRUCTIUNEA MOVE

Pina acum am facut transferuri de date de la un program catre un canal sau invers. Instructiunea MOVE va permite sa mutati date de la un canal la altul. De exemplu, pentru a muta date de la claviatura la ecran, introduceti:

```
10 MOVE #1 TO #2
```

apoi:

```
RUN
```

Orice veti introduce de la claviatura va apare pe ecran. Dar veti descoperi ca apasarea lui BREAK nu face decit sa tipareasca un spatiu pe ecran. Pentru a iesi din aceasta capcana, apasati CR pina ce ajungeti la ultima linie din ecran. Apoi, raspundeti cu BREAK la intrebarea scroll? (Ar trebui ca pe viitor sa evitati sa mutati date de la claviatura la orice alta cale pentru ca s-ar putea sa nu mai reusiti sa iesiti din instructiunea MOVE.)

Instructiunea MOVE se mai poate utiliza si pentru a examina fisierele memorate pe floppy. De exemplu, daca mai aveti pe floppy fisierul "Numere" (vezi pagina 7), ii puteti examina continutul cu instructiunea:

```
10 MOVE "d";1;"Numere" TO #2
```

(Remarcati ca nu trebuie sa deschideti/inchideti (OPEN/CLOSE) fisierul, MOVE face singur lucrul acesta.)

Deasemenea, pentru a face o copie a programului "Numere" introduceti:

```
10 MOVE "d";1;"Numere" TO "d";1;"Numere 2"
```

In acest caz, MOVE deschide o cale pentru a citi din fisierul existent ("Numere") si o alta cale pentru a scrie in fisierul nou creat ("Numere 2"). Apoi citeste datele din "Numere" si le scrie in "Numere 2". Apoi inchide ambele cai.

MOVE va functiona atit cu numere de cale (ca de ex. #4), cit si cu specificatoare de canale (ca de ex. "d";1;"Numere"). Caille standard #1, #2 si #3 nu pot fi insa specificate cu numele consacrate K, S si P.

Puteti face o copie de siguranta a fisierului "Numere" pe alt disc folosind:

```
10 MOVE "d";1;"Numere" TO "d";2;"Numere 2"
```

Instructiunea MOVE poate fi folosita si pentru a trimite fisiere catre imprimanta. Daca aveti o imprimanta legata la interfata seriala, introduceti:

```
10 FORMAT "t",viteza  
20 OPEN #4,"t"  
30 MOVE "d";1;"Numere" TO #4
```



### Programul Printer Server

Programul permite unui HC85 cuplat la o retea sa controleze o imprimanta seriala. Imprimanta poate fi folosita de toate calculatoarele cuplate la retea. Acest program este util daca, de exemplu, un grup de utilizatori de HC85 poseda o singura imprimanta seriala pe care vor sa o imparta. Se arata totodata o utilizare mai deosebita pentru instructiunea MOVE.

Calculatorul folosit ca Printer Server trebuie sa fie intotdeauna statia 64, si trebuie intotdeauna sa faca legatura cu statia 62 (care este o statie speciala de stabilire de contact). Astfel, statia emitatoare foloseste temporar statia 62, si trimite numarul sau real de statie, de la care va muta apoi un fisier spre canalul t. Pentru a stabili un program Printer Server introduceti:

```
10 FORMAT "n";64
20 OPEN #4;"n";62: INPUT #4;a$: CLOSE #4
30 MOVE "n";CODE a$ TO "t"
40 OPEN #4;"b": PRINT #4;CHR$ 12: CLOSE #4: RUN
```

(Linia 40 trimite un avans de pagina.)

Programul de mai jos este cel folosit de emitator. Mai intii, emitatorul stabileste temporar statia 62. Apoi se emite numarul real de statie al emitatorului. Apoi statia emitatorului revine la numarul sau real. In final, linia 60 trimite datele care trebuiesc imprimate (in acest caz listingul).

```
10 LET statia=numar
+----- introduceti aici numarul de
+----- statie al HC-ului local
20 FORMAT "n";62
30 OPEN #4;"n";64: PRINT #4;CHR$ statia: CLOSE #4
40 FORMAT "n";statia
50 OPEN #4;"n";64
60 LIST #4
70 CLOSE #4
```

### INSTRUCTIUNILE CLEAR # SI CLS #

Se recomanda folosirea instructiunilor CLS # si CLEAR # in prima linie a oricarui program care foloseste interfata 1.

### Instructiunea CLEAR #

Asa cum instructiunea CLEAR sterge toate variabilele definite (operatie care se executa in mod automat si la RUN), instructiunea CLEAR # sterge toate canalele si caile definite prin program, efectuind urmatoarele operatii:

- decupleaza toate caile de la canalele deschise de catre utilizator
- elibereaza spatiul de memorie ocupat de aceste canale. (Zona CHANS va contine numai canalele predefinite "k", "s" si "p".)
- cupleaza caile #0, #1, #2 si #3 la canalele standard.
- trece toate discurile in starea R/W (vezi mai jos).



Nu trebuie sa se confunde efectul instructiunii CLEAR # cu efectul inchiderii prin CLOSE # a tuturor cailor. Spre deosebire de CLOSE #, instructiunea CLEAR # abandoneaza pur si simplu datele care se gasesc in canale. Daca, spre exemplu, se sterge prin CLEAR # un canal de disc prin care s-a scris intr-un fisier, datele din ultimul buffer vor fi pierdute, si mai grav, nici datele care au fost deja scrise pe disc nu vor fi accesibile pentru citire.

Mesajul de eroare "Disk 'R/O'" si CLEAR #  
Pentru a proteja datele inscrise pe floppy, interfata 1 utilizeaza o metoda de a preveni erorile datorate schimbarilor incorecte de floppy.

Daca interfata detecteaza o schimbare de suport intr-unul din minidrive-urile cu care a lucrat de la ultimul NEW sau CLEAR #, ea trece in mod automat discul respectiv in modul 'R/O' (numai citire).

Daca dupa o astfel de schimbare se incearca o operatie care necesita scrierea de date pe acel disc, se va obtine mesajul de eroare "Disk 'R/O'".

Pentru a corecta aceasta situatie, trebuie executata instructiunea CLEAR # imediat inainte de orice schimbare de suport intr-unul din Minidrive-uri.

#### Instructiunea CLS #

Efectele acestei instructiuni sint similare cu executia comenzilor:

PRINT;; BORDER 7: PAPER 7: INK 0: CLS

#### JOCUL DE RETEA

Anexa 1

Pe discul de demonstratie livrat odata cu sistemul exista o copie a acestui joc. Numele de fisier este "net game". Programul este un bun exemplu pentru utilizarea retelei. Parti din el pot fi utile si in programe scrise de dumneavoastra.

#### Jocul

Pentru a juca acest joc, cei doi parteneri trebuie sa se gindeasca fiecare la un numar intre 1 si 100. Cistigatorul jocului este cel care ghiceste primul numarul adversarului. La fiecare tentativa, calculatorul va va spune cit de aproape sinteti.

#### Programul

Subrutina de la linia 500 decide cine este utilizatorul 1 si cine este utilizatorul 2. Asta este necesar pentru ca atunci cind se transmit tentativele unul dintre voi foloseste subrutina de la linia 1100, iar celalalt subrutina de la linia 1200, si astfel utilizatorul 1 trimite primul, iar utilizatorul 2 primeste primul.

Programul decide cine este utilizatorul 1 trimitind catre celalalt calculator mesajul "1", si apoi intrind in ascultare pe retea. Daca primeste un "1", asta inseamna ca celalalt HC85 a pornit programul mai tirziu. Primul HC85 trimite de aceea un "2" catre calculatorul adversarului, si se face singur utilizatorul 1. Daca, pe de alta parte, programul primeste inapoi un "2", asta inseamna ca celalalt program era deja pornit si in asteptare atunci cind programul local a trimis "1". Programul local se face singur utilizatorul 2.



Daca cele doua programe pornesc in acelasi timp, cele doua mesaje "1" se vor ciocni pe retea, programele se vor bloca amandoua in asteptare, si este necesara intreruperea unuia dintre programe cu BREAK. si restartarea.

Programul principal schimba numele utilizatorilor, citeste numarul secret (care nu este trimis adversarului) si apoi compara tentativele. Mai intii se transmite tentativa si apoi se afiseaza raspunsul.

Liniiile de la 190 incolo detecteaza o victorie, o afiseaza corespunzator si apoi ofera un alt joc.

```
10 GO SUB 500
20 PRINT: BORDER 1: PAPER 1: INK 7: CLS
30 PRINT "Joc de ghicit numere" "Introduceti mai intii un
   numar secret, apoi ghiciti-l pe al adversarului"
40 INPUT "Cum va numiti?";a$
50 PRINT "Salut ";a$
60 GO SUB 1000+100*user
70 PRINT "Jucati cu ";b$
75 PRINT "a$,b$"
80 INPUT "Ghinditi-va la un numar (1 la 100)";a
90 IF a<1 OR a>100 OR a<>INT a THEN GO TO 80
130 INPUT "Ce numar incercati?";b
140 LET a$=STR$ b: GO SUB 1000+100*user

150 LET c=ABS (a-VAL b$)
160 IF c=0 THEN LET a$="Asta este": GO TO 170
161 IF c<4 THEN LET a$="Arde": GO TO 170
162 IF c<10 THEN LET a$="Fierbinte": GO TO 170
163 IF c<20 THEN LET a$="Foarte cald": GO TO 170
164 IF c<40 THEN LET a$="Cald": GO TO 170
165 IF c<60 THEN LET a$="Rece": GO TO 170
166 LET a$="Gheata"
170 GO SUB 1000+100
180 PRINT b$,a$
190 IF c=0 OR b$="Asta este" THEN GO TO 210
200 GO TO 130
210 IF b$="Asta este" THEN PRINT FLASH 1;"Victorie":
   FOR n=0 TO 7: BORDER n: BEEP .1,n: BEEP .1,n+16:NEXT n:
   GO TO 230
220 PRINT "Infringere": FOR n=7 TO 0 STEP -1: BORDER n:
   BEEP .2,n: NEXT n
230 BORDER 1: INPUT "Alt joc? (d/n)";a$
240 IF a$="d" THEN RUN 20
250 STOP
500 OPEN #4;"n":0
510 PRINT #4;"1"
520 CLOSE #4
530 OPEN #4;"n":0
540 INPUT #4;a$
545 CLOSE #4
550 IF a$="1" THEN OPEN #4;"n":0: PAUSE 5: PRINT #4;"2":
   LET user=1
560 IF a$="2" THEN LET user=2
570 CLOSE #4
580 FORMAT "n":user: RETURN
1100 OPEN #4;"n":3-user
1110 PRINT #4;a$
1120 CLOSE #4
1130 OPEN #4;"n":3-user
1140 INPUT #4;b$
1150 CLOSE #4
```

PROLOG — DIALOG — EPILOG



```

1160 RETURN
1200 OPEN #4;"n";3-user
1210 INPUT #4;b$
1220 CLOSE #4
1230 OPEN #4;"n";3-user
1240 PRINT #4;a$
1250 CLOSE #4
1260 RETURN

```

# VARIABLE DE SISTEM

## Anexa 2

Pe linga variabilele de sistem tabelate in sectiunea 3.22 a Manualului tehnic pentru HC85, interfata 1 utilizeaza urmatoarele variabile:

Tip	Adresa	Nume	Continut
X1	23734	FLAGS3	Biti de control interfata 1
X2	23735	VECTOR	Adresa folosita pentru a extinde interpretorul BASIC
X10	23737	SBRT	Rutina de paginare a ROM-urilor
2	23747	BAUD	Numar pe 16 biti care determina rata de transfer pe linia seriala calculata astfel: $BAUD = (3500000 / (26 * \text{baud rate})) - 2$ O puteti folosi pentru a stabili viteze nestandard de comunicatie seriala.
1	23749	NTSTAT	Numarul statiei locale pe retea
1	23750	IOBORD	Bitii 2..0 contin culoarea marginii ecranului in timpul I/E prin IF1. Puteti pune orice culoare doriti cu instructiunea POKE.
N2	23751	SER_FL	Spatiu de lucru de 2 octeti pentru interfata seriala.
N2	23753	SECTOR	2 octeti nefolositi
N2	23755	CHADDT	Salvare pentru indicator caracter curent
1	23757	NTRESP	Locatie folosita pentru raspuns in retea
1	23758	NTDEST	Inceput bloc de control in retea. Contine numarul statiei destinatie 0-64.
1	23759	NTSRCE	Numarul statiei sursa
X2	23760	NTNUMB	Numarul blocului 0-65535
N1	23762	NTTYPE	Tip bloc 0-normal 1-ultimul (EOF).
X1	23763	NTLEN	Lungime bloc de date 0-255
N1	23764	NTDCS	Suma de control pentru blocul de date
N1	23765	NTHCS	Suma de control pentru blocul de control
N2	23766	D_STR1	Inceputul primului specificator 8 octeti Contine numar minidrive 0-2 pe 16 biti
N1	23768	S_STR1	Contine numar cale 0-15
N1	23769	L_STR1	Tip dispozitiv "D", "N", "T" sau "B"
N2	23770	N_STR1	Lungime nume fisier
N2	23772	F_STR1	Adresa nume fisier
N8	23774	D_STR2	Al doilea specificator de 8 octeti folosit de MOVE si LOAD.
N1	23782	HD_00	Inceput zona de lucru pentru SAVE, LOAD, VERIFY si MERGE: cod tip de date 0=prog, 1=numere, 2=sir, 3=cod
N2	23783	HD_0B	Lungime bloc de date 0-65535
N2	23785	HD_0D	Adresa in memorie a blocului 0-65535
N2	23787	HD_0F	Lungime program fara variabile
N2	23789	HD_11	Numarul liniei de autostart
1	23791	COPIES	1 octet nefolosit
	23792		Inceputul zonei CHANS
	23813		Inceputul programului BASIC cu IF1 activat dar fara canale utilizator



## OBSERVATII

1. Inserarea variabilelor de sistem se efectueaza in mod automat la prima aparitie a unei erori, a unei comenzi specifice interfetei 1 sau in cazul mesajului OK. Aceasta inserare poate genera mesajul Out of memory daca cei 58 de octeti necesari nu sint disponibili.
2. Deschiderea unei cai sau a unui canal de disc sau retea necesita o anumita cantitate de memorie. Un canal de disc are 306 octeti, iar un canal de retea are 276. Aceste canale vor fi create fie prin OPEN # sau prin MOVE. Daca RAMTOP este prea jos, aceste comenzi pot genera mesajul de eroare Out of memory.
3. Un alt efect al introducerii variabilelor de sistem sau al crearii canalelor este mutarea programelor in cod masina aflate in instructiuni REM. Puneti intotdeauna aceste programe dupa RAMTOP.

### CANALUL DE DISC

### Anexa 3

La fiecare deschidere a unui fisier prin una din instructiunile OPEN # sau MOVE, in zona denumita CHANS in manualul de BASIC se creaza o zona de memorie denumita canal. De obicei un canal este adresat in limbaj masina de registrul IX. Canalul are o lungime de 306 octeti si contine un bufer de 256 de octeti.

Continutul canalului este urmatorul:

0	Adresa 8
2	Adresa 8
4	'D' sau 'D'+80H pentru un canal ad-hoc
5	Adresa rutinei de iesire din ROM-ul din IF1
7	Adresa rutinei de intrare din ROM-ul din IF1
9	Lungime canal, adica 306
11	CHFLAG 0=citire, 1=scriere posibila din/in acest canal
12	CHDRIVE numar drive folosit de canal 0=curent, 1=1, 2=2
13	CHNAME Numele fisierului completat cu spatii pina la 11 caractere, octet 9 bit 7=r/o, octet 10 bit 7=sys
24	20 de octeti folositi de sistemul de gestiune fisiere
44	CHCR Numar inregistrare curenta in extensia curenta
45	CHRR0 Numar inregistrare pentru a acces aleator
47	CHRR2 Indicator depasire capacitate fisier in acces aleator
48	CHBYTE Indicator caracter curent in buferul de date
50	CHDATA 256 octeti pentru bufer

Deschiderea unui canal de disc nu creaza o harta de ocupare in memoria BASIC. Hartile de ocupare disc exista in permanenta in memoria RAM instalata pe IF1, memorie comutata impreuna cu ROM-ul din IF1.

### CANALUL DE RETEA

La deschiderea unei cai catre retea se creeaza o zona de memorie denumita canal in spatiul indicat de variabila de sistem CHANS. Aceasta zona este adresata in limbaj masina de registrul IX. Canalul are o lungime de 276 octeti si contine un bufer de 255 de octeti.

Continutul canalului este urmatorul:

PROLOG — DIALOG — EPILOG



0	Adresa 8
2	Adresa 8
4	'N' pentru OPEN # sau 'N'+80h pentru MOVE
5	Adresa rutinei de iesire din ROM-ul din IF1
7	Adresa rutinei de intrare din ROM-ul din IF1
9	Lungime canal adica 276
11	NCIRIS Numarul statiei partenerie in comunicatie
12	NCSELF Numarul statiei locale la deschiderea canalului
13	NCNUMB Numarul blocului 0-65535
14	NCTYPE Tipul pachetului de date... 0=normal,1=ultimul
15	NCOBL Numarul de octeti in blocul de date
17	NCDCS Suma de control pentru blocul de date
18	NCHCS Suma de control pentru blocul de control
19	NCCUR Deplasament in bufer pentru ultimul caracter transferat
20	NCIBL Numarul de octeti utili din bufer
21	NCB 255 octeti pentru buferul de date

### CANALUL DE SERIALA

La deschiderea unei cai catre interfata seriala se creeaza o zona de memorie denumita canal in spatiul indicat de variabila de sistem CHANS. Aceasta zona este adresata in limbaj masina de registrul IX. Canalul are o lungime minima de 11 octeti.

Continutul canalului este urmatorul:

0	Adresa 8
2	Adresa 8
4	'B' sau 'T'
5	Adresa rutinei de iesire din ROM-ul din IF1
7	Adresa rutinei de intrare din ROM-ul din IF1
9	Lungime canal adica 11

### CONEXIUNI INTERFATA SERIALA

Anexa 4

Conectorul de interfata seriala este folosit precum urmeaza:

1. Neconectat
2. TXData (intrare)
3. RXData (iesire)
4. DTR (intrare) trebuie sa fie la nivel ridicat pentru 'gata'
5. CTS (iesire) este la nivel ridicat daca este 'gata'
6. Neconectat
7. Masa
8. Neconectat
9. +12v

Pentru conectare cu o interfata standard CCITT V24, in capatul calalalt al cablului trebuie folosit un conector cu 25 de pini cablat in felul urmator:

2. TXData
3. RXData
5. CTS
6. +12v (USR)
7. Masa
20. DTR



Instructiunile implementate de interfata 1 genereaza mesaje de eroare diferite de mesajele de eroare generate de ROM-ul din placa de baza. Aceste mesaje vor fi urmate de numarul liniei si numarul comenzii din linie care a generat eroarea.

Aceste noi mesaje de eroare sint listate mai jos in ordine alfabetica:

CODE error

Ati incercat sa incarcati (LOAD) un bloc de cod a carui lungime este mai mare decit lungimea specificata de instructiunea LOAD.

Disk error

In timpul executiei unei operatii de intrare/iesire pe disc a aparut o eroare care nu a putut fi reparata prin reincercari

Disk full

Ati incercat sa scrieti date intr-un disc care nu avea suficient spatiu liber. Reincercati programul cu un alt disc, sau eliberati spatiu pe discul curent stergind fisierele de care nu mai aveti nevoie.

Disk 'R/O'

Ati incercat sa efectuati o operatie de scriere pe un suport schimbat, fara sa comunicati calculatorului prin CLEAR # faptul ca ati terminat de lucrat cu vechiul suport. Introduceti CLEAR # si apoi repetati comanda.

Disk 'write' protected

Ati incercat o operatie de scriere pe un disc care are montata protectia la scriere. Indepartati protectia si apoi reincercati.

File not found

Ati incercat o operatie asupra unui fisier inexistent, sau ati incercat o operatie CAT pe un disc fara nici un fisier.

File 'R/O'

Ati incercat sa stergeti sau sa scrieti un fisier care are atributul de protejat la scriere (octetul 9 din nume bitul 7=1). Deprotejati fisierul daca sinteti sigur ca vreti sa-l modificati.

Invalid device expression

Ati specificat un dispozitiv diferit de k, s, p, d, n, b sau t. Acelasi mesaj se obtine daca ati folosit punct-virgula in loc de virgula pentru unul din specificatorii k, s sau p.

Invalid drive number

Ati specificat un numar de Minidrive mai mare ca 2, sau ati specificat numarul 0 (Minidrive-ul curent), inainte de a-l declara printr-un apel explicit.

Invalid name

Numele fisierului este fie un sir vid, fie are mai mult de unsprezece caractere.

Invalid station number

S-a specificat un numar de statie in afara domeniului 0-64 (1-64 pentru instructiunea FORMAT).



#### Invalid stream number

Numarul de cale specificat este in afara domeniului 0-15.

#### MERGE error

Ati incercat sa comasati date sau cod. MERGE functioneaza numai cu programe.

#### Missing baud rate

Lipseste rata de transfer in instructiunea FORMAT "b" sau "t"

#### Missing drive number

Lipseste numarul minidrive-ului

#### Missing name

Lipseste numele fisierului

#### Missing station number

Lipseste numarul statiei in retea

#### Program finished

Ati incercat sa executati o linie dincolo de ultima linie din program. Acest mesaj de eroare va apare daca executati un GO TO urmat de un numar de linie mai mare decit ultima linie din program. Va apare deasemenea daca introduceti RUN fara a avea un program in memorie.

#### Reading a 'write' file

Incercati sa cititi date dintr-un fisier disc inexistent, sau dintr-un canal care a fost deja folosit pentru scriere.

#### Stream already open

Ati incercat sa deschideti o cale care a mai fost folosita pentru un canal de tip nou (d, n, t sau b). Calea poate fi deschisa numai dupa ce a fost inchisa.

#### Verification has failed

Exista diferente intre fisierul salvat si programul, datele sau codul existente in memorie.

#### Writing to a 'read' file

Ati incercat sa scrieti intr-un fisier disc existent. Fisierul existent trebuie mai intii sters, daca nu este nevoie de el. Altfel trebuie utilizat un fisier nou.

#### Wrong file type

Ati incercat sa incarcati (LOAD) un fisier de date sau cod ca pe un program sau invers un program ca pe un fisier de date sau cod.

### BASIC-UL EXTINS

#### Anexa 6

Interfata 1 extinde BASIC-ul existent deja in HC85. Extensiile si adaugarile sint rezumate mai jos.

#### Calle

Calle sint specificate prin #n unde n este un numar in domeniul 0-15. Calle 0, 1, 2 si 3 sint de obicei folosite de BASIC. Caracterul # este parte din cuvintul cheie pentru instructiunile OPEN # si CLOSE #.



### Canalele

Exista sapte tipuri de canale in BASIC-ul extins: claviatura (k), ecranul (s), imprimanta (p), interfata seriala pentru texte (t), interfata seriala binara (b), retea (n) si discul (d).

Fiecare canal este specificat prin litera lui care poate fi majuscula sau nu. Reteaua si discul au nevoie de informatii suplimentare pentru a specifica complet canalul.

Un canal de retea necesita un numar de statie, asa incit un specificator de retea are forma "n";x unde x este numarul statiei in domeniul 0-64.

Un canal de disc necesita un numar de minidrive si un nume de fisier, asa incit un specificator de disc are forma "d";y;"nume" unde y este un numar de minidrive in domeniul 0-2 iar nume este un sir cu 1 pina la 11 caractere.

### Minidrive curent

Primul minidrive la care se face acces dupa NEW sau CLEAR # va deveni ceea ce se numeste minidrive-ul curent. Din acest moment inainte acest minidrive poate fi specificat si cu numarul 0.

Daca se incearca folosirea specificatorului 0 inainte de a defini minidrive-ul curent, se va semnala eroarea Invalid drive number.

### Caracterul ? in nume de fisier

Este bine sa nu folositi caracterul ? in numele de fisiere, pentru ca acest caracter are alt rol. El este folosit de sistemul de fisiere pe post de 'Jolly Jocker' (wild card), putind sa inlocuiasca orice alt caracter, dar unul si numai unul.

Daca de exemplu trebuie sa stergeti fisierele cu numele "nume0", "nume1", "nume2" si nu aveti alte fisiere cu nume de forma "numex" (unde x este orice caracter), puteti folosi o singura comanda de forma ERASE "d",1,"nume?" care sterge toate cele trei fisiere.

Daca intr-un OPEN # specificati partial un nume de fisier existent (folosind caracterul "?") va fi folosit primul fisier al carui nume se potriveste.

### Instructiuni

#### CAT y

Listeaza toate numele fisierele aflate pe floppyul din minidrive-ul y. Lista este prezentata in ordinea din catalogul discului si este urmata de dimensiunea spatiului liber pe disc in kiloocteti. ATENTIE ! Daca discul a fost schimbat si nu s-a introdus NEW sau CLEAR #, numarul reprezentind spatiul liber nu reflecta realitatea.

#### CAT #z;y

Trimite catalogul floppy-ului din minidrive-ul y catre calea z.

#### CAT y;"cc...c"

Listeaza numele fisierele care se potrivesc cu sirul de caractere "cc...c", care poate contine "?" pentru specificari ambigue.



<u>CAT #z;y;"cc...c"</u>	Ca mai sus, dar trimite lista catre calea z.
<u>CLEAR #</u>	Readuce sistemul de cai si canale la starea de dupa NEW - exista numai canalele standard k, s, p si sint deschise numai caile standard #0, #1, #2 si #3. Eventualele date existente in canalele utilizator sint ignorate, spatiul de memorie fiind eliberat fara remuscari.
<u>CLOSE #cale</u>	Desface legatura dintre calea specificata si orice canal. Daca exista date blocate prin scriere in buferul canalului atunci acestea sint fie transmise (pe retea) sau inregistrate (pe floppy).
<u>CLS #</u>	Readuce ecranul in starea de dupa NEW. BORDER alb, PAPER alb, INK negru, ecran sters.
<u>ERASE "d";y;"nume"</u>	Sterge fisierele specificate de nume aflate pe discul din minidrive-ul y. Numele poate contine caracterul "?" pentru specificatii ambigue.
<u>FORMAT "d";y</u>	Pregateste un floppy din minidrive-ul y pentru a fi utilizat din BASIC.
<u>FORMAT "n";x</u>	Stabileste numarul statiei pe retea la x.
<u>FORMAT "t";x</u> <u>FORMAT "b";x</u>	Stabileste viteza de comunicatie pentru interfata seriala la x (x trebuie ales dintre vitezele standard de comunicatie 50, 110, 300, 600, 1200, 2400, 4800, 9600, 19200).
<u>INKEY##cale</u>	Intoarce un singur caracter sub forma unui sir daca cel putin unul este disponibil sau intoarce sirul yid "" daca nu exista caracter disponibil din calea respectiva.  Aceasta instructiune are sens doar daca calea este legata la un canal de retea sau de interfata seriala.
<u>INPUT#cale;var</u>	Citeste variabila var din calea specificata. Calea trebui sa fi fost deschisa inainte catre un canal de intrare. Este important sa retineti ca orice element de PRINT care apare in instructiunea INPUT va fi scris catre aceasta cale. Aceasta este de obicei necesar numai atunci cind se citesc date de la claviatura. Retineti de asemenea ca separatorul "," scrie un caracter.  Optiunea LINE este disponibila ca mai inainte.



**LOAD \*canal optiuni** Incarca programul, datele sau codul de la canalul specificat. Se pot folosi numai canalele "b", "n" sau "d".

Toate optiunile existente pentru LOAD sint disponibile si la LOAD \*.

**MERGE \*canal optiuni** La fel ca LOAD, doar ca nu sterge liniile de program sau variabilele decit pentru a face loc pentru unele noi cu acelasi numar de linie sau nume.

**MOVE sursa TO destinatie** Muta datele de la sursa catre destinatie. Sursa si destinatia pot fi numere de cale sau canale.

Comanda se termina numai la intilnirea unui indicator de sfirsit de fisier in sursa: aceasta se poate intimpla doar daca sursa este un canal de retea sau disc, sau altfel o cale legata la un astfel de canal.

Daca sursa sau destinatia sint specificate drept canale, atunci acestea sint deschise la inceput si inchise la terminarea transferului.

**OPEN #cale, canal** Leaga calea specificata la canalul specificat pentru a permite programului BASIC sa citeasca sau sa scrie din/in acel canal. Calea trebuie sa fie inchisa sau deschisa catre unul din canalele k, s sau p.

**PRINT #cale....** Tipareste secventa de PRINT catre calea specificata. Calea trebuie sa fi fost deschisa in prealabil catre un canal de iesire.

Secventa de PRINT poate avea aceeasi sintaxa ca mai inainte si poate contine alte elemente de tipul #cale.

**SAVE \*canal optiuni** Salveaza programul, datele sau codul catre canalul specificat. Pot fi folosite numai canalele "b", "n" sau "d".

Toate optiunile existente la SAVE sint disponibile si la SAVE \*.

**VERIFY \*canal opt** La fel ca LOAD (vezi mai sus) cu exceptia faptului ca datele nu sint incarcate in memorie, ci sint doar comparate cu ceea ce exista deja acolo.



## APELURI DE SISTEM

Acest manual descrie modalitatea standard de apel a functiilor Interfetei 1 din limbaj de asamblare. El este destinat programatorilor mai avansati, care doresc sa utilizeze limbajul de asamblare, pentru a creste viteza programelor si/sau pentru a realiza functii neimplementate la nivel de BASIC.

Inainte de a citi acest manual, este necesara parcurgerea manualului de utilizare a interfetei din limbajul BASIC. Acest manual este doar o completare a manualului sus mentionat.

Desi exista practic posibilitatea de a comuta PROM-ul Interfetei 1 peste PROM-ul din placa de baza, si apoi de a apela direct rutine din PROM-ul Interfetei 1, aceasta modalitate nu este recomandata pentru ca o modificare ulterioara a programelor din Interfata 1 va duce la incompatibilitatea vechilor programe cu noua interfata.

Un anume apel de sistem, si anume apelul de sistem care permite accesul la gestiunea fisierelor este descris foarte sumar in acest manual. Pentru detalii suplimentare se recomanda manualul de interfata BDOS, manual care descrie in detaliu functiile sistemului de gestiune a fisierelor.

Modalitatea de apel a rutinelor din interfata 1 este urmatoarea: se foloseste o instructiune RST 8 urmata de un octet care identifica tipul functiei cerute. Dupa executia rutinei cerute, executia programului continua cu octetul care urmeaza codului rutinei.

De exemplu apelul:

```
RST      8
DEFB     31H      ; octet denumit 'cod de apel'
...        ; executia continua aici
```

insereaza variabilele sistem extinse.

(Datorita modului de transfer al parametrilor catre apelurile de sistem, acest apel de creare a variabilelor extinse trebuie sa fie primul apel de sistem.)

Codurile apelurilor de sistem sint alocate dupa codurile de eroare (codul -1 reprezinta mesajul 'OK', 0 este 'NEXT without FOR', 1 este 'var not found', s.a.m.d.).

Valorile pentru codurile apelurilor sint:

-1..26	rezulta mesajele de eroare din masina de baza
27..32	coduri valide compatibile Sinclair
33..44	coduri invalide (microdrive la Sinclair)
45..50	coduri valide compatibile Sinclair
51..58	coduri valide compatibile numai HC85
59..254	coduri invalide

Daca se incearca un cod invalid se va obtine mesajul de eroare:

Hook code error



'Hook code' este denumirea consacrată în literatura Sinclair pentru 'apel de sistem'.

### TRANSFERUL PARAMETRILOR

Dacă apelul de sistem are un singur parametru de tip octet, acesta se transferă în general prin acumulator. La fel dacă rezultatul rutinei este un singur octet, acesta este întors în acumulator.

Dacă rutina are mai mulți parametri, aceștia se transferă prin locații de memorie specifice fiecărui apel, locații plasate în variabilele extinse.

Nar înainte de a putea modifica o variabilă extinsă, aceasta trebuie să existe în memorie, de aceea se impune apelul de creare a variabilelor extinse ca prim apel de sistem.

### TRATAREA ERORILOR

Erorile care apar în executia instrucțiunilor BASIC duc la întreruperea programului, abandonarea canalelor ad-hoc și afișarea unui cod de eroare.

Dacă apar erori în timpul executiei unui apel de sistem, codul de eroare este plasat în variabilă de sistem ERRNR, și apoi se încarcă registrul SP cu conținutul variabilei de sistem ERRSP, după care se execută un RETURN. Modificând variabilă de sistem ERRSP, utilizatorul poate și trebuie să preia tratarea erorii.

#### Exemplu de tratare a erorilor:

##### RUTINA\_APEL\_IF1:

```
LD HL, (ERRSP)
PUSH HL ; salvez vechi ERRSP
LD HL, TRATARE_EROAARE
PUSH HL ; adresa rutinei de tratare
LD (ERRSP), SP ; modific stiva de eroare
RST 8
DEFB cod_apel
; continuare normala
POP HL ; descarc rutina de eroare
POP HL
LD (ERRSP), HL ; refac vechea stiva de eroare
RET
```

##### TRATARE\_EROAARE:

```
POP HL
LD (ERRSP), HL ; refac vechea stiva de eroare
LD HL, ERRNR
LD A, (HL) ; preiau codul erorii
LD (HL), -1 ; pun OK în loc
RET ; tratare efectiva eroare
; retur în programul care a
; apelat RUTINA_APEL_IF1
```

```
ERRSP EQU 5C3DH
ERRNR EQU 5C3AH
```



In timpul executiei unui apel de sistem sint posibile urmatoarele erori:

<u>Cod Mesaj standard</u>	<u>Descriere</u>
---------------------------	------------------

- |                                     |  |
|-------------------------------------|--|
| <u>3. Out of memory</u>             | Spatiu insuficient pentru creare variabile extinse sau canal.  |
| <u>4. Invalid drive number</u>      | Ati incercat sa utilizati drive-ul curent (cod 0) inainte de a-l defini printr-un apel explicit.   |
| <u>12. Writing to a 'read' file</u> | Ati incercat sa scrieti prin RST 10H intr-un canal configurat pentru citire (numai pentru "r" sau "d").  |
| <u>13. Reading a 'write' file</u>   | Ati incercat sa cititi prin CALL 15E6H dintr-un canal configurat pentru scriere (numai pentru "r" sau "d").  |
| <u>14. Disk 'write' protected</u>   | Ati incercat sa scrieti pe un disc care avea montata protectia la scriere.   |
| <u>15. Disk full</u>                | Nu mai exista spatiu disponibil in catalogul sau in zona de date a discului selectat.  |
| <u>16. Disk error</u>               | Eroare hard pe discul selectat.  |
| <u>17. File not found</u>           | La inchiderea unui fisier nu a mai fost gasita intrarea din catalog pusa la deschiderea fisierului. Probabil ati schimbat discul inainte de a inchide toate fisierele deschise pentru scriere. |
| <u>20. BREAK into program</u>       | Ati apasat CAPS SHIFT/SPACE in timpul executiei unui apel de sistem.   |
| <u>23. Disk 'R/O'</u>               | Disc protejat soft. Apelati BDOS functia 0 dupa orice schimbare de suport!   |
| <u>24. File R/O'</u>                | Fisier protejat. Ati incercat sa stergeti un fisier protejat, sau ati creat un fisier cu atributul de protectie setat.   |

In continuare sint prezentate apelurile de sistem grupate dupa dispozitivele la care se leaga.



## INTERFATA SERIALA

Citire consola (Console Input)

cod: 27 i: - o: A = caracter

Asteapta apasare tasta, si intoarce codul ASCII al tastei.

Scriere la consola (Console Output)

cod: 28 i: A = caracter o: -

Trimite caracter catre calea #2 (ecran partea de sus).

Stare consola (Console Status)

cod: 32 i: - o: CY = stare claviatura

Intoarce CY=1 daca exista cel putin o tasta apasata, altfel CY=0.

## CONSOLA

Citire de la seriala (Input from RS232)

cod: 29 i: - o: A = caracter, CY = 1 sau  
A = 0, CY = 0 pentru time-out  
e: 20

Receptioneaza un caracter pe interfata seriala la viteza data de variabila BAUD (5CC3H).  $BAUD = (3500000 / (26 * viteza)) - 2$

Pentru a semnala faptul ca HC85 este gata sa primeasca date, ridica semnalul CTS. Dupa receptia unui caracter, sau dupa time-out, coboara semnalul CTS.

Daca dupa primirea unui caracter, si dupa coborirea semnalului CTS, se mai receptioneaza un alt caracter - CTS nu s-a propagat suficient de repede si partenerul a mai inceput un alt caracter - se receptioneaza si acest caracter. El este memorat in variabila SER\_BYTE (5CC8H), iar SER\_FL (5CC7H) este setat pentru a semnala prezenta acestui caracter catre apelul urmator de citire de la seriala.

Daca la apelul urmator SER\_FL este setat, se va intoarce imediat caracterul din SER\_BYTE, dupa stergerea lui SER\_FL. In acest caz semnalul CTS nu mai este comutat.

In timpul asteptarii si receptiei, marginea ecranului are culoarea data de variabila IOBORD (5CC6H).

Scriere la seriala (Output to RS232)

cod: 30 i: A = caracter o: - e: 20

Rutina asteapta indefinit activarea semnalului DTR, care semnaleaza faptul ca partenerul transferului este gata sa receptioneze date. Asteptarea poate fi intrerupta prin BREAK.

Cei 8 biti din acumulator sint apoi emisi (bitul 0 primul), urmati de 1 bit de stop.

Viteza de emisie este stabilita de variabila BAUD, iar culoarea marginii de variabila IOBORD.



## IMPRIMANTA

Sciere la imprimanta (Output to printer)

cod: 31 i: A = caracter o: -

Caracterul din acumulator este trimis catre calea #3, care este de obicei cuplata cu un canal care trimite datele la imprimanta.

## RETEA

Deschide canal (Open net ad-hoc)

cod: 45 i: DSTR1 = # statie amic 0..64, cuvint la 5CD6H  
NTSTAT = # statie locala 1..64, cuvint la 5CC5H  
o: IX = adresa canal  
e: 3

Funcția rezerva și initializează un canal de rețea, dar nu face nici o operație de i/e.

Inchide canal (Close net ad-hoc)

cod: 46 i: CURCHL = adresa canal, o: - e: -

Conform convențiilor standard, adresa canalului ar fi trebuit să se transmită în registrul IX, dar o eroare în ROM-ul din interfața 1 de la Sinclair, face ca adresa să se preia de fapt din variabila CURCHL. Din motive de compatibilitate, am păstrat această 'jumătate' de eroare.

Rutina emite ultimul buffer prin rețea, dacă s-a scris în acest canal și dacă există date în buffer, adică NCORL > 0. Apoi eliberează spațiul de memorie ocupat de canal.

Recepție pachet rețea (Receive Net Packet)

cod: 47 i: IX = adresa canal, o: Cy=eroare e: -

Rutina recepționează un pachet pe rețea, emis de NCIRIS către NCSTAT, cu numărul de pachet dat de NCNUMB. Dacă pachetul este recepționat corect se întoarce cu Cy=0 și NCNUMB incrementat.

În cazul în care NCIRIS=0 se așteaptă o emisie 'broadcast', adică se așteaptă la infinit (sau pînă la BREAK). Dacă NCIRIS>0, atunci așteptarea este cu time-out (12 milisec). Chiar dacă pachetul sositeste în ultimul moment și are lungimea maximă, rutina nu trebuie să dureze mai mult de 50 milisec.

Flagul Cy este setat în caz de time-out sau în cazul apariției oricărei alte erori: stație sursă sau destinație necorespunzătoare, număr pachet eronat, sume de control eronate.

Emisie pachet rețea (Send net packet)

cod: 48 i: IX = adresa canal, A = EOF; o: - e: -

Acumulatorul conține 0 pentru un pachet normal, sau 1 pentru ultimul pachet. NCORL conține numărul de octeți utili din buffer. Dacă NCIRIS=0 nu se așteaptă achitare.

Dacă emisia se termină fără eroare, NCNUMB este incrementat.



## DISCUL

Deschide canal (Open disk ad-hoc)

```
cod: 51  i: DSTR1 = drive 0..2,      cuvint la 5CD6H
          FSTR1 = adresa nume 0..65534,  cuvint la 5CDCH
          NSTR1 = lungime nume 1..11     cuvint la 5CDAH
          o: IX = adresa canal
          e: 3, 4, 14, 15, 16, 23
```

Deschide in zona CHANS un canal de disc, initializeaza FCB-ul cu parametrii luati din DSTR1, FSTR1 si NSTR1, apoi deschide fisierul. Daca nu gaseste fisierul in director, creaza un fisier nou si seteaza bitul 0 din CHFLAG.

Inchide canal (Close disc channel)

```
cod: 52  i: IX = adresa canal  o: -
          e: 4, 14, 15, 16, 17, 23, 24
```

Serie ultimul buffer pe disc (daca CHFLAG este setat iar CHBYTE este diferit de zero), si apoi inchide FCB-ul. Apoi elibereaza spatiul memorie ocupat de canal.

Citire secventiala (Read sequential)

```
cod: 53  i: IX = adresa canal
          o: CY = 0, Z = 0 sfirsit fizic fisier (BDOS)
             CY = 0, Z = 1 sfirsit logic fisier (A = 1AH)
             CY = 1, A = primul octet din buffer
          e: 16
```

Se citeste urmatorul sector (256 octeti) din fisier, se seteaza corespunzator CHBYTE, dupa care se preia primul octet din buffer.

Pentru a obtine urmatoarele caractere din buffer se forteaza variabila CURCHL (5C51H) la adresa canalului, si apoi se apeleaza rutina 15E6H.

Pentru a citi caracter cu caracter un fisier sint necesari urmatoorii pasi:

- deschidere canal disc
- memorare IX in CURCHL
- citire secventiala (care intoarce primul caracter sau EOF)
- apel 15E6H pentru caracterele urmatoare, pina la Cy=0 (EOF)

Scriere secventiala (Write sequential)

```
cod: 54  i: IX = adresa canal  o: -
          e: 14, 15, 16, 17, 23, 24
```

Se scrie buferul in fisier, se sterge octetul high din CHBYTE.

Creare canal (Make disc channel)

```
cod: 55  i: DSTR1 = drive 0..2,
          FSTR1 = adresa nume 0..65534,
          NSTR1 = lungime nume 1..11.
          o: IX = adresa canal
          e: 3
```



Deschide in zona CHANS un canal de disc, initializeaza FCB-ul cu parametrii luati din DSTR1, FSTR1 si NSTR1, dar spre deosebire de apelul 51 NU deschide fisierul.

Se poate utiliza pentru a creea un canal de disc care va fi folosit direct cu apeluri BDOS.

Distrugere canal (Kill channel)

cod: 56 i: IX = adresa canal o: -

Elibereaza spatiul de memorie ocupat de canal, actualizind variabilele de sistem (rutina RECLAIM) si zona STREAMS.

Apelul poate fi folosit pentru oricare din canalele standard IF1 (cele care au lungimea data de octetii 9 si 10), adica pentru canalele B, T, D si N.

### BDOS

Exista o portita de intrare directa catre sistemul de gestiune a fisierelor pe disc. Acesta este un subset al sistemului BDOS din CP/M. Transferul parametrilor este diferit, codurile functiilor sint diferite, dar principial sint aceleasi functii.

Erorile soft (No dir space) sint tratate identic ca in CP/M, adica se intorc coduri in acumulator. Erorile ireparabile (Bad sector, Drive R/O, File R/O, Select) sint tratate prin mecanismul cu ERRSP (vezi mai sus).

BDOS call

cod: 57 i: HD11 = argument cuvint la 5CEDH  
COPIES = numar functie octet la 5CEFH  
o: A = cod retur BDOS

Numar	Argument	Rezultat	Descriere
0	-	-	initializare DOS
1	0..1	-	selectie disc
2	.fcb	dir cod	deschide fisier
3	.fcb	dir cod	inchide fisier
4	.fcb	dir cod	cauta primul
5	-	dir cod	cauta urmatorul
6	.fcb	dir cod	sterge fisier
7	.fcb	0 = ok	citire secventiala
8	.fcb	0 = ok	scriere secventiala
9	.fcb	dir cod	creare fisier
10	.fcb	dir cod	redenumire fisier
11	-	login	intoarce vector login
12	-	-1..1	intoarce disc curent
13	DMA	-	stabileste adresa transfer DMA
14	-	.alloc	intoarce adresa vector alocare ATENTIE! este in 'phantom RAM'
15	-	-	protejeaza soft discul curent
16	-	R/O vec	intoarce vector protectie discuri
17	.fcb	dir code	stabileste attribute fisier
18	.fcb	cod err	citire randomiala
19	.fcb	cod err	scriere randomiala
20	.fcb	r0,r1,r2	calculeaza dimensiune fisier
21	.fcb	r0,r1,r2	stabileste adresa acces aleator
22	logoff	-	decupleaza drive-uri selectate
23	.fcb	cod err	scriere randomiala cu 'fill'

Pentru detalii suplimentare vezi manual interfata CP/M.



## RWTS

cod: 58    i: HD11 = adresa bloc i/o    cuvint la 5CEDH  
o: Acc = 0 daca a mers ok  
e: -

### Rutina efectueaza patru functii distincte:

- formare disc
- citire sector
- scriere sector
- pornire motor si pozitionare fara verificare

In caz de eroare, rutina face un numar rational de reincercari.

### Formatul blocului de intrare iesire este:

<u>Deplasament</u>	<u>Descriere</u>
0	tip bloc (trebuie sa fie 01)
1	numar drive (0 sau 1)
2	numar volum (trebuie sa fie 0)
3	numar pista (0..39 sau 0..34)
4	numar sector (0..15)
5	adresa DMA (16384..65535)
7	adresa bufer extins (trebuie sa fie 2932H)
9	adresa tabela caracteristici drive (std 1F2AH)
11	comanda 0 = start motor + pozitionare 1 = citire sector 2 = scriere sector 4 = formare disc (toate piste)
12	rezultat operatie 00H = ok 08H = nu pot forma discul 10H = disc protejat 20H = eroare volum 40H = eroare drive 80H = eroare citire (dupa reincercari)
13	numarul de volum citit din cimpul de identificare
14..18	spatiu de lucru al rutinei RWTS

Utilizatorul trebuie sa completeze octetii 0..11. Octetii 12..18 vor fi completati de rutina RWTS ca rezultat al operatiei.

Si in cazul formarii se foloseste buferul de date: primul octet este multiplicat in ceilalti 255 de octeti, iar bufferul rezultat este in scris in fiecare sector format pe disc.

### Formatul tabelii de caracteristici:

<u>Deplasament</u>	<u>Descriere</u>
0	tip dispozitiv (trebuie sa fie 01)
1	intirziere acces pista la pista in milisec.
2	timpe stabilizare pe pista in milisec.
3	timpe pornire motor in sutimi de secunda
4	adresa tabela intratesere sectoare (std 1F30H)

Tabela de intratesere sectoare contine 16 octeti folositi pentru a obtine numarul fizic al sectorului logic dorit.



## ACCES DIRECT LA RUTINELE DIN IF1

Daca se doreste accesul direct la rutinele sau datele din memoria fantoma (ROM si RAM) se poate folosi urmatorul mecanism:

```

LD HL, CIUDATENIE
LD (HD11), HL
RST 8
DEFB 50

CIUDATENIE:
; DE AICI INCOLO IN SPATIUL
; 0..16383 SE VEDE MEMORIA FANTOMA
; ??? VA UREZ SUCCES !
; DEZACTIVARE MEMORIA FANTOMA

...
CALL 700H
EQU 5CEDH

HD11

```

## INDICE NUMERIC APELURI DE SISTEM

Cod	Intrare	Iesire	Descriere
27	-	A=car	citire consola
28	A=car	-	scriere consola
29	-	A=car+Cy A=0 Cy=0	citire seriala time out
30	A=car	-	scriere seriala
31	A=car	-	scriere imprimanta
32	-	Cy=1	daca exista contact la tastatura
33	A=drive	-	invalid (select microdrive)
34	-	IX=adr	invalid (open microdrive)
35	IX=adr	-	invalid (close microdrive)
36	-	-	invalid (delete microdrive file)
37	IX=adr	-	invalid (read seq microdrive)
38	IX=adr	-	invalid (write seq)
39	IX=adr	-	invalid (read random)
40	IX=adr	-	invalid (read sector)
41	IX=adr	-	invalid (read next sector)
42	IX=adr	-	invalid (write sector)
43	IX=adr	-	invalid (open microdrive)
44	IX=adr	-	invalid (delete channel)
45	DSTR1	IX=adr	deschide canal retea DSTR1=#statie amic
	NTSTAT	-	NTSTAT=#statie propriu
46	CURCHL=adr	-	inchide canal retea
47	IX=adr	Cy=err	citeste pachet retea
48	IX=adr	-	emisie pachet retea
49	-	-	creere variabile de sistem
50	HD11=adr	-	acces direct la memoria fantoma
51	DSTR1	IX=adr	deschide canal disc; DSTR1=drive
	FSTR1	-	FSTR1=adr nume
	NSTR1	-	NSTR1=lungime nume
52	IX=adr	-	inchide canal disc
53	IX=adr	A=car Cy, Z	citeste urmatorul sector; Cy=1 este ok Cy=0, Z=0 EOF BDOS, Cy=0, Z=1 EOF 1AH
54	IX=adr	-	scrie sector
55	DSTR1	IX=adr	creaza canal disc; DSTR1=drive
	FSTR1	-	FSTR1=adr nume
	NSTR1	-	NSTR1=lungime nume
56	IX=adr	-	distrug canal
57	HD11	A=rez	apel BDOS; HD11=argument
	COPIES	-	COPIES=functie
58	HD11	A=rez	apel RWTS; HD11=adresa bloc i/e



## UTILIZAREA MEMORIEI

Variabilele sistem extinse se insereaza la prima executie a unei instructiuni RST 8 de catre interpretorul BASIC din masina de baza (rutina care incepe la adresa 0008H este rutina standard de eroare din sistem). Acest apel poate fi generat din mai multe motive:

- o eroare reala (inclusiv STOP sau OK)
- o instructiune din setul extins (care nu este acceptata de analizorul BASIC din masina de baza)
- un 'apel de sistem'

Inserarea variabilelor sistem deplaseaza in memorie programul BASIC, variabilele, etc., deci si programele in limbaj masina din instructiunile REM. Se impune plasarea acestor programe dupa RAMTOP, sau inainte de variabilele sistem (in ecran sau in buferul de imprimanta).

La deschiderea unei cai, in zona CHANS se creeaza un canal, localizat la sfirsitul zonei, impingind programul, variabilele, zona ELINE, etc. spre adrese mari (rutina MAKE\_ROOM). Pointerii de sistem sint actualizati corespunzator. Adresa canalului este memorata in zona sistem denumita STREAMS.

Creearea unui canal nu modifica adresele altor canale. Dar totusi, canalele pot fi mutate in memorie la inchiderea unuia din canalele deschise: toate canalele care urmeaza canalului de inchis sint mutate impreuna cu PROG, VARS, etc. catre adrese mici. Din acest motiv canalele nu contin adrese absolute, ci numai deplasamente locale.

Singurele modificari de adrese care se fac la inchiderea unui canal sint localizate in variabilele sistem PROG, VARS, ELINE, etc. si zona STREAMS.

Situatia este oarecum simplificata fata de Interfata 1 de la Sinclair, pentru ca la HC85 vectorii de alocare ai spatiului pe disc sint ascunsi in memoria RAM fantoma, si cuplarea logica a unui drive nu mai implica crearea spatiului de memorie necesar prin impingerea zonei CHANS (zona MAPS nu mai exista).

Canalele deschise de utilizator prin apeluri de sistem sint alocate tot la sfirsitul zonei CHANS, dar sistemul nu are cum sa informeze utilizatorul de eventualele schimbari de adrese (datorate unor inchideri de canale). Se impune, din acest motiv, urmatoarea regula:

Deschideti si inchideti canalele BASIC si utilizator intr-o stricta ordine FIFO.

Numai in acest fel se poate garanta, faptul ca adresa canalului nu se schimba in timp.

Canalele deschise de utilizator prin apeluri de sistem sint marcate setind bitul 7 din octetul 4 (octet care contine litera care identifica tipul canalului). La fel sint marcate canalele generate de instructiunea MOVE. Aceste canale poarta numele de canale 'ad-hoc'.



Toate canalele ad-hoc au viata scurta. Ele sint abandonate de sistem la prima revenire in mod comanda, deci si la STOP sau OK. ATENTIE, abandonate si nu inchise: sistemul nu face altceva decit sa elibereze spatiul de memorie ocupat de aceste canale.

O secventa corecta de apeluri de sistem trebuie deci sa inchida canalele cu care a lucrat, inainte de a se reveni in mod comanda.

## ORGANIZAREA LOGICA A DISCULUI

### Glosar de termeni:

#### SECTOR:

cantitatea minima de informatii care poate fi schimbata intre un program si suportul disc. Sectoarele sint 16 pe o pista numerotate 0 la 15, si au cite 256 de octeti fiecare.

#### PISTA:

cicumferinta pe suportul magnetic. Sint 35 sau 40 numerotate de la 0 la 34 respectiv 39. Capacitatea unei piste este de  $16 \times 256 = 4k$ .

#### FATA:

una din suprafetele suportului magnetic. Discul are doua fete utile care insa nu pot fi accesate simultan (discul trebuie intors de pe o fata pe alta). Capacitatea unei fete este deci de  $35 \times 16 \times 256 = 140k$  respectiv  $40 \times 16 \times 256 = 160k$ .

#### SPATIU SISTEM:

zona de disc care incepe de la primul sector de pe prima pista si care se extinde un numar intreg de piste. Contine sistemul de operare CP/M. La HC85 zona sistem nu exista pentru ca sistemul de operare se gaseste in ROM.

#### SPATIU DE DATE:

zona de disc care incepe imediat dupa zona sistem si continua spre interiorul discului pina la ultimul sector. Contine catalogul discului plus datele din fisiere plus spatiul disponibil.

#### GRUP:

unitatea de alocare a spatiului pe disc, folosita atit pentru catalog cit si pentru fisiere. La HC85 are dimensiunea minima, adica 1 kilooctet (4 sectoare). Grupurile sint numerotate incepind de la 0 pina la dimensiunea spatiului de date minus unu ( $139 = 8BH$  respectiv  $159 = 9FH$ ).

#### CATALOGUL:

zona de la inceputul spatiului de date care contine numele fisierelor aflate pe disc si alte date (atribute, dimensiune, alocare, etc.) Catalogul ocupa la HC85 doua grupuri (G00 si G01) adica 2 kiloocteti, ceea ce lasa 138 respectiv 158 kiloocteti pentru datele fisierelor.

#### EXTENSIE:

unitatea de alocare spatiu in catalog (32 octeti). La HC85 sint  $2048 : 32 = 64$  de intrari in catalog. Structura unei extensii este data mai jos. O extensie este utilizata



pentru a indica grupurile folosite de un fisier (maxim 16).  
Daca un fisier ocupa mai mult de 16 grupuri, atunci el ocupa  
mai multe extensii, numerotate incepind cu 0.

Un fisier ocupa spatiu atit in catalog, cit si in spatiul  
disponibil pe disc.

Din spatiul disponibil, un fisier ocupa un numar intreg de  
grupuri, chiar daca ramin citeva sectoare nefolosite la coada  
ultimului grup.

In catalog un fisier ocupa atitea intrari cite sint necesare  
pentru a indica catre toate grupurile pe care le ocupa.

Ultima extensie a unui fisier se recunoaste prin faptul ca  
octetul 15 este mai mic ca 128=80H. Toate extensiile  
intermediare au acest octet egal cu 80H. Acest octet reprezinta  
de fapt numarul de sectoare de 128 octeti (stilul CP/M) ocupate  
efectiv de fisier in extensia respectiva (128x128=16k).

#### Structura unui extensii este:

<u>Deplasament</u>	<u>Descriere</u>
0	cod utilizator 00..0F sau E5 pentru fisier sters
1..11	bitii 6..0 = nume fisier extins cu blancuri
1..8	bitul 7 = atribut utilizator
9	bitul 7 = atribut R/O (numai citire)
10	bitul 7 = atribut SYS (invizibil)
11	bitul 7 = atribut rezervat
12	bitii 4..0 = numarul extensiei
13..14	alti indicatori
15	contor sectoare logice (128 octeti) in aceasta extensie
16..31	16 locatii pentru maxim 16 grupuri ocupate de extensia curenta

#### STRUCTURA FISIERELOR

Fisierele disc sint de doua tipuri: fisiere text, care pot fi  
exploatare cu OPEN #, INPUT #, PRINT # sau CLOSE #, si fisiere  
program generate cu SAVE \* si citite prin LOAD \*.

Fisierele text contin incepind chiar din primul octet caracterele  
care au fost scrise in canalul asociat. Liniile sint separate  
doar prin intoarcere car (CR). Ultimul sector este umplut cu  
caracterul 1AH care este indicatorul de sfirsit de fisier text.

Fisierele program au doua parti: un header de 9 octeti si apoi  
efectiv datele salvate. Headerul are urmatoarea structura:

<u>Deplasament</u>	<u>Descriere</u>
0	tip: 0=program, 1=numere, 2=siruri, 3=cod
1	lungime date: 0-65535
3	adresa date: 0-65535 (utila numai la tip=3)
5	lungime program fara variabile 0-65535 (tip=0) identificator scurt pentru variabila (tip=1..2)
7	numarul liniei de lansare program (numai tip=0) 0-9999, daca nu are start automat bit 15 = 1







● Volumul al doilea cuprinde: VI. Microcalculatorul HC-85 în procese industriale. Pachete de programe generalizabile pe HC-85. VII. Calculatorul personal în învățămînt și educație. VIII. Programe educaționale în BASIC pe HC-85. IX. Microbibliotecă de programe pe casete X. Complemente de matematică. XI. Complemente informatică. XII. Anexe, volumul finalizînd cunoașterea calculatorului și a limbajelor universale BASIC și LOGO printr-o impresionantă cantitate de programe, prin noi aprofundări și informații consistente.

● Se adresează unor cercuri largi de cititori, atît școlari din clasele I-VIII, elevi din clasele IX-XII, studenți, cadre didactice

de specialitate precum și muncitori, tehnicieni și alte categorii de ierarhi din toate domeniile economiei naționale care folosesc calculatoare personale în proiectare-producție sau în activitatea de instruire-educare-specializare.

● Cartea poate fi utilizată cu mult succes de cei care au acces la numeroasele calculatoare HC-85 sau compatibile instalate în unități de cercetare-proiectare-producție, în unitățile de învățămînt, în centre și tabere de instruire, la domiciliu, fiind totodată utilă tuturor celor care nu dispun încă de asemenea calculatoare, dar au nevoie sau doresc să învețe sistematic structura, funcționarea și, mai ales, utilizarea lor.

